Empirical studies on word representations

Suster, Simon

Empirical studies on word representations

Simon Šuster

The work presented here was carried out under the auspices of the Center for Language and Cognition Groningen (CLCG) at the Faculty of Arts of the University of Groningen.

Groningen Dissertations in Linguistics 154
ISSN: 0928-0030
ISBN: 978-90-367-9261-5 (printed version)
ISBN: 978-90-367-9260-8 (electronic version)

Document prepared with LaTeX 2ε and typeset by pdfTeX (TeX Gyre Pagella font)
Cover design: Anna Kravcov and Simon Šuster
Cover image: Netherlandish Proverbs, Pieter Bruegel the Elder, 1559
Printed by: Wöhrmann Thesis

rijksuniversiteit
groningen

Empirical studies on word representations

Proefschrift

ter verkrijging van de graad van doctor aan de
Rijksuniversiteit Groningen
op gezag van de
rector magnificus prof. dr. E. Sterken
en volgens besluit van het College voor Promoties.

De openbare verdediging zal plaatsvinden op

donderdag 10 november 2016 om 16.15 uur

door

Simon Šuster

geboren op 10 april 1983
te Maribor, Slovenië

Promotor
Prof. dr. G. J. M. van Noord

Beoordelingscommissie
Prof. dr. P. Hendriks
Prof. dr. A. Søgaard
Prof. dr. P. T. J. M. Vossen

Acknowledgements

I'd like to acknowledge and thank several people who were important during the four years I spent working on this thesis. Fore and foremost, I'm indebted to my supervisor Gertjan van Noord. I've been extremely fortunate to receive his invaluable professional and personal support, and the freedom to pursue my research ideas. I appreciate all the time he devoted to our (sometimes long) weekly discussions, and his pragmatic and critical attitude to scientific research. I learned a lot about becoming a researcher from him. Next, I'd like to thank Ivan Titov for investing time and effort in working with me. The work in the second half of my PhD has been greatly shaped by his mentorship. I learned a lot from him, and I truly admire his knowledge and the speedy insights into problems. Thanks also go to the assessment committee, Petra Hendriks, Anders Søgaard and Piek Vossen, who read this thesis and provided useful comments.

After studying computational linguistics as an LCT master student for two years, one spent in Groningen and another in Nancy, I was convinced that I would like to continue—or rather, only truly begin—doing research in this exciting field. I'd presumed back then that particularly Groningen would be a great place to work. And so it was. Many people I've encountered here made my work and stay more pleasant: Barbara, Çağrı, Daniel, Dieke, Floris, Gosse, Gregory, Harm, Johan, Johannes, John, Kilian, Leonie, Malvina, Noortje, Martijn, Rob, Valerio. It was a pleasure sharing the office with Angelina, Daniel, Dieke, Jakolien, Jin and Proscovia.

I'd also like to thank Gorus and Marijke for promptly and professionally resolving all my questions and requests; Jan-Wouter for entertaining chitchats about Slovenian alpine skiing and his support for my research visit abroad; to the people at the high-performance computing center for efficient reactions to my numerous requests. I also appreciate the support of Nvidia Corporation with the donation of a GPU unit for this research.

Teaching and assisting were a valuable experience—thanks to Gertjan, Gosse and Malvina for giving me this opportunity. Special thanks go to my paranymphs Malvina and Rob for helping with the organization of the defense ceremony.

I've been fortunate to spend parts of the PhD abroad, namely in the CCL group of Leuven and the CLiPS group of Antwerp. They wholeheartedly accepted me as a visiting researcher and made me feel extremely welcome. I'd like to thank Frank, Ineke, Liesbeth, Tom and Vincent from Leuven; and the cheerful bunch from Antwerp: Walter, Ben, Chris, Enrique, Giovanni, Guy, Janneke, Mike, Robert, Stéphan and the "girls from above".

Since PhD is luckily not only work, I'd like to thank the following people, who have also become friends: Natasha, Erik and Darja; Julija, Marcel and the kids; Jeffrey, Mami, Miwa and Towa; Marvin, Zelda and Kevin; Çağrı, Joanna and Franek; Valerio and Sara—we all spent some really precious moments together! Thanks also to Nicola for his generous hospitality during the first days in Groningen, to Johan H. for his help and entertainment, and to our warm-hearted neighbors Toppie, Hans and Annemarie.

Big thanks to my family on the other side of the Alps for moral and practical support. Finally, I'm immensely grateful to Artyom, Enej and Olga for going through this adventure together and standing by through all the ups and downs. Thanks for your patience and encouragement!

Contents

Acknowledgements **v**

Contents **vii**

List of Figures **xii**

List of Tables **xiv**

1 Introduction **1**

I Background **9**

2 Representing words **11**
2.1 Overcoming lexical sparseness 14
2.2 Applicability of word representations 18
2.3 Formulating the representation learning problem 19

3 Selected approaches to word representations **23**
3.1 Concept-based semantic classes 24
3.2 Clustering and probabilistic latent-variable models 26
3.3 (Neural) word embeddings 30

4 Applications for word representations **35**
4.1 Intrinsic applications 37

4.1.1 Semantic similarity 37
4.1.2 Wordnet-based similarity 39
4.2 Extrinsic applications 41
4.2.1 Named-entity recognition 41
4.2.2 Part-of-speech tagging 42
4.2.3 Syntactic parsing 44
4.2.4 Semantic frame identification 46

II Reducing lexical sparseness with semantic classes 49

5 Using wordnet concept classes in a grammar-driven parser 51
5.1 Motivation 52
5.2 Experimental setup 54
5.2.1 Corpora 58
5.2.2 Statistical analysis of parser's performance and disambiguation model lookup 58
5.2.3 Lexical semantic inventory 59
5.2.4 Disambiguation method 61
5.2.5 Levels of semantic representation 63
5.2.6 Application of semantic classes to disambiguation model 63
5.3 Empirical study 64
5.3.1 Relationship between parsing accuracy and disambiguation model lookup 64
5.3.2 Generalization with semantic classes 68
5.4 Additional related work 71
5.5 Conclusion 73

III The role of syntax in models of word representations 77

6 Dependency Brown clustering 79
6.1 Motivating the use of syntactic context 79
6.2 The Brown et al. (1992) clustering 81

6.3 The extension with a dependency language model 84
6.4 Experimental setup for Dutch 86
6.5 Empirical study for Dutch 87
6.5.1 Cluster examples 88
6.5.2 Cluster quality 90
6.5.3 Amount of data 93
6.5.4 Data selection with dependency relations 95
6.6 A study on English 97
6.6.1 Experimental setup 97
6.6.2 Empirical summary 98
6.7 Additional related work 100
6.8 Conclusion 101

7 A discrete latent-variable model with syntactic functions 103
7.1 Hidden Markov models and their variants 104
7.2 Motivating syntactic functions 105
7.3 A tree model with syntactic functions 106
7.4 Learning and inference 111
7.4.1 State splitting and merging 112
7.4.2 Decoding for HMM-based models 113
7.5 Empirical study 116
7.5.1 Parameters and setup 116
7.5.2 Data for obtaining word representations 120
7.5.3 Evaluation tasks 120
7.5.3.1 Named entity recognition 120
7.5.3.2 Semantic frame identification 121
7.5.4 Baseline word representations 122
7.5.5 Preparing word representations 123
7.5.6 NER results 125
7.5.7 SFI results 127
7.5.8 Further discussion 128
7.6 Additional related work 129
7.7 Conclusion and future work 132

IV Learning word representations in a multilingual context 135

8 Bilingual learning of multi-sense embeddings with discrete autoencoders 137

8.1 Introduction . . . 137
8.2 Contributions . . . 139
8.3 Word embeddings with discrete autoencoders . . . 140
8.3.1 Learning and regularization . . . 142
8.3.2 Obtaining word representations . . . 144
8.4 Word affiliation from alignments . . . 144
8.5 Parameters and set-up . . . 145
8.5.1 Learning parameters . . . 145
8.5.2 Bilingual data . . . 146
8.6 Evaluation tasks . . . 147
8.6.1 Word similarity . . . 147
8.6.2 Supersense similarity . . . 148
8.6.3 POS tagging . . . 148
8.7 Results . . . 149
8.7.1 Overview . . . 149
8.7.2 The amount of (parallel) data . . . 153
8.7.3 The dimensionality and frequent words . . . 153
8.7.4 The number of senses . . . 154
8.8 The effect of second language . . . 154
8.8.1 The importance of bilingual signal . . . 154
8.8.2 The effect of language choice . . . 157
8.9 Additional related work . . . 162
8.10 Conclusion and future work . . . 164

V Conclusion 167

9 Summary and conclusions 169

Appendices 175

A Dependency Brown clustering objective **177**
A.1 Simplifying the objective function 177

B Sum-product message passing **181**
B.1 Background 181
B.2 Message passing 182

C Individual semantic similarity results from Chapter 8 **187**

Bibliographical abbreviations **193**

Bibliography **195**

Nederlandse samenvatting **219**

Groningen dissertations in linguistics (GRODIL) **223**

List of Figures

2.3 Representation learning as part of supervised machine learning. 21

3.1 Example of concept classes and hypernymy in English WordNet 25
3.2 A Brown clustering tree induced from Dutch tweets. 28
3.3 Illustration of HMM representations. 30
3.4 Analogy in word embeddings. 32

4.1 Evaluation types from the thesis. 36
4.2 A sentence annotated with named entities. 41
4.3 A sentence annotated with parts of speech. 44
4.4 Dependency structure for an English sentence. 45
4.5 Constituent structure for an English sentence. 45
4.6 Parsing as a two-stage process. 46
4.7 An example frame-semantic parse. 47

5.1 Example of an incorrect parse. 53
5.2 Obtaining intermediately-grained representations. 62
5.3 In-model information and parsing accuracy (per sentence). . . . 65
5.4 In-model information and parsing accuracy (per instance). . . . 66

6.1 An illustration of a clustering tree hierarchy. 82
6.2 Effect of bit string length. 92
6.3 Effect of amount of data. 94

7.1 A Hidden Markov tree model with syntactic functions. 107
7.2 Comparing the shape of transition probability distributions. . . 110
7.3 Pseudo-counts in a tree HMM model with syntactic functions. . 112
7.4 A 2D visualization of representations learned with the model using syntactic functions. 115
7.5 Convergence of batch and online training regimes in HMM learning. 117
7.6 Convergence in learning of sequential and tree HMMs. 119

8.1 An autoencoding model. 140
8.2 Effect of amount of data used in learning on the SCWS correlation scores. 152
8.3 Effect of embedding dimensionality 153
8.4 Importance of second-language context. 155
8.5 Effect of second-language window size. 156
8.6 Effect of language choice on POS tagging. 161

B.1 Message passing on a tree. 183

List of Tables

5.1 Alpino feature types. 55
5.2 Cornetto statistics per part-of-speech category. 60
5.3 10 most common semantic types in Cornetto. 60
5.4 In-model proportions of partially instantiated feature types. . . 67
5.5 A shortlist of partially instantiated feature types with respective statistics. 68
5.6 Parsing results obtained by generalization with semantic classes. 69
5.7 Parsing result for a single partially instantiated feature type. . . 70

6.1 Example dependency clusters. 89
6.2 Comparison of similarity scores for the standard and the dependency Brown clustering. 90
6.3 Comparison of similarity scores for the standard and the dependency Brown clustering, varying parameters. 91
6.4 Effect of data selection with dependency relations. 96
6.5 Comparison of similarity scores for the standard and the dependency Brown clustering for English. 98

7.1 The syntactic functions selected for the final runs. 124
7.2 NER results for English and Dutch. 125
7.3 NER results for the English MUC dataset. 127
7.4 Frame identification results for English. 128

8.1 List of parallel corpora. 146

8.2 Results for all evaluation tasks. 150
8.3 Comparison to other works. 151
8.4 Results of the model trained with the infinite second-language window. 157
8.5 Effect of language choice on semantic similarity. 159

C.1 Individual results: RU-EN. 188
C.2 Individual results: CZ-EN. 188
C.3 Individual results: FR-EN. 189
C.4 Individual results: ES-EN (NC). 189
C.5 Individual results: DE-EN (NC). 190
C.6 Individual results: RU-EN (NC). 190
C.7 Individual results: CZ-EN (NC). 191
C.8 Individual results: FR-EN (NC). 191

CHAPTER 1

Introduction

The goal of natural language processing (NLP) is the modeling of human language from a computational perspective. By modeling, we have in mind building computational approaches that can learn, understand and generate natural language, just as we humans do. NLP largely relies on some theory of human languages, and at the same time, it can contribute to the understanding of the human language through its analyses. The main focus of NLP are difficult, real-life problems such as machine translation, spoken dialogue, and analysis of vast amount of online data, for example by mining social media for information about health or about attitude of consumers towards products (Hirschberg and Manning, 2015). Whatever the end application of NLP, analyses on several linguistic levels are usually required, including morphological, syntactic, semantic and pragmatic levels. What makes these tasks hard (especially for the computer) is that the ambiguity is prevalent in human language, and it often requires deep knowledge about the world to be properly resolved.

The work presented in this thesis focuses on the lexical semantic part of analysis. It addresses the question of how we can represent words so that we can meaningfully compare them computationally and so that the representations can be of value in solving practical NLP problems. The word

representations can be obtained in several ways; they can be crafted by humans or learned from corpora. The central idea common to the studies of all types of word representations[1] is *generalization*:

> If the task is simply to remember accurately a set of unrelated items, the generalization effects are harmful and are called interference. But generalization is normally a helpful phenomenon. It allows us to deal effectively with situations that are similar but not identical to previously experienced situations.

This quote from Rumelhart et al. (1986) in "Parallel Distributed Processing", their seminal, cognitive-science oriented work on neural network methods, captures well the very purpose of using word representations. Although the kind of representations Rumelhart et al. discuss in their work is defined to include only the distributed, i.e. dense low-dimensional representations, we believe that the above description of generalization applies to the study of all types of word representations. In processing of language, new, previously unseen situations constantly arise; this is true even for models built on tremendously large amounts of textual data (we expand on this in Chapter 2). Yet, with the aid of word representations, we can make decisions in an unseen situation by relating it to a similar situation that *was* encountered previously. This is possible because word representations allow us to generalize, i.e. relate similar words to each other.

All in all, we work with four distinct methods of obtaining word representations. Each corresponds to a thesis chapter. A more precise diagram of correspondences between the representation methods, NLP problems and chapters can be found in Figure 4.1 on page 36. Although we explore the methods mostly with different research questions and evaluation tasks in mind, we gain an understanding of advantages and disadvantages of each of these methods on a higher level as well. We mention these as the discussion unfolds throughout the following chapters, and we reflect upon

[1] And in NLP more generally, see for example Daelemans and van den Bosch (2005).

them succinctly in the concluding section. We now address the organization of the thesis and the research questions more specifically.

The thesis is thematically organized in four parts. There are three parts consisting of altogether four chapters in which we present empirical studies involving word representations. These main parts are preceded by Part I, in which we lay the theoretical groundwork for the subsequent parts. In each of the three parts, we deal with a different research question, and work within distinct word representation frameworks. More concretely, in Part II we are asking ourselves the following question:

Q1 Do word representations obtained from a human-built lexical inventory effectively resolve the lexical sparseness problem in syntactic parsing?

Compared to the parts that follow, the approach here is unique in that the focus is not on learning word representations as these are already given, but rather on how to apply the information in a purposeful way to a specific parser for Dutch. Part III that follows represents the largest unit of the thesis. Here, we estimate the word representations from data, and specifically study the role of syntax in learning, particularly in contrast to techniques that rely on word sequences. The research question in this part is therefore:

Q2 How can a syntax-based definition of word context lead to better word representations?

We examine two ways of incorporating syntactic information into models of word representations: the first is to include only bare syntactic structures, thereby transforming the linear definition of context predominant in distributional learning (Chapter 6); the second includes the syntactic structure together with syntactic functions (labels), in line with a hypothesis that word contexts play different roles depending on their syntactic functions (Chapter 7). In these chapters, two different approaches are used to induce word representations, namely word clusters and latent-variable models, which lead to word representations with quite different characteristics,

although the approaches themselves are theoretically related, and the second can be seen as an extension of the first. Along the way, we also examine the distinction between generic representations (one per word type) and context-dependent, multi-sense representations, which can represent each textual occurrence of the same word type differently. In Part IV, we go on to propose a method for learning multi-sense representations in the word embedding framework. The main emphasis in this part can be phrased as:

Q3 Can bilingual, parallel corpora be employed for better multi-sense word representations?

Thus, we use supervisory signal available from parallel corpora (translations) in building a sense predictor for a given language, and use that predictor in the learning of multi-sense word embeddings for the same language. Just as in the previous part, these representations have the potential to assign distinct word representations depending on the context. Compared to other parts of the thesis in which word representations are studied only monolingually involving either English or Dutch, we approach the representation learning in Part IV from a multilingual perspective.

Chapter guide

Chapter 2 lays the background on lexical semantics with a focus on distributional definition of meaning and polysemy. We motivate the use of word representations through discussion and illustration of lexical sparseness. We introduce representation learning from a theoretical standpoint, and position the word representation learning in the larger endeavor of supervised machine learning. Chapter 3 introduces the various frameworks for learning word representations, but limiting the overview only to those four used in this thesis. Chapter 4 presents an overview of intrinsic and extrinsic applications intended to measure the quality of word representations. Similarly, we focus only on those employed in the thesis. We introduce each application including an example, describe the available datasets and motivate the use of word representations in that application.

Chapter 5 presents an empirical study of lexical sparseness in a syntactic parser for Dutch. We address the issue specifically by altering the parsing component that relies on bilexical preferences for disambiguation. The concept classes obtained from a Dutch wordnet are applied at different levels of generality, and the effect on the parser's accuracy is measured. **Q1**

Chapter 6 deals with distributional learning of word representations with a word clustering approach. We take the well-known Brown clustering algorithm and adapt its sequential nature to work with dependency structures obtained with a parser. We investigate the quality of the redefined algorithm by evaluating against human-built wordnet classes for Dutch and English. We find the approach to work especially well for Dutch, with the advantages observed across different possible parametrizations of the clustering method. In Chapter 7, we include syntactic functions as an additional source of information available to the model. We use variants of sequential and tree-structured hidden Markov models to induce word representations. We discuss how such models—in contrast to the previous chapter—provide context-sensitive representations through a soft relationship between a word type and its semantic class(es). We explore the models' flexibility regarding inference and decoding. We evaluate the models extrinsically and show that the model with syntactic functions—when compared to other hidden Markov models, the clustering representations from the previous chapter, as well as word embeddings—is advantageous mostly for Dutch, but not for English. **Q2**

Chapter 8 builds on a well-known embedding model to account for polysemy and situate the learning in a multilingual context. We present an autoencoding architecture which jointly captures latent sense distribution and learns the parameters of the embedding model. We use word-aligned parallel corpora as a signal that can help to estimate a more reliable sense predictor in our model. We find that our approach is beneficial on tasks measuring various kinds of semantic similarity. On an extrinsic POS tagging task, we obtain mixed results. We also examine the effect of the identity and the family of the second language used in learning. **Q3**

Publications

This thesis is based on the following publications:

- Šuster, S. and van Noord, G. (2013). Semantic mapping for lexical sparseness reduction in parsing. In *ESSLLI'13 Workshop on Extrinsic Parse Improvement*.
- Šuster, S. and van Noord, G. (2014). From neighborhood to parenthood: the advantages of dependency representation over bigrams in Brown clustering. In *COLING*.
- Šuster, S., van Noord, G., and Titov, I. (2015). Word representations, tree models and syntactic functions. *arXiv preprint arXiv:1508.07709*.
- Šuster, S., Titov, I., and van Noord, G. (2016). Bilingual learning of multi-sense embeddings with discrete autoencoders. In *NAACL-HLT*.

Some parts may also refer to the following:

- Šuster, S. (2012). Resolving PP-attachment ambiguity in French with distributional methods. Master's thesis, University of Groningen and University of Lorraine.
- Hürlimann, M., Weck, B., van den Berg, E., Šuster, S., and Nissim, M. (2015). GLAD: Groningen Lightweight Authorship Detection. In *Working Notes Papers of the CLEF 2015 Evaluation Labs*.
- Šuster, S. (2015). An investigation into language complexity of World-of-Warcraft game-external texts. *arXiv preprint arXiv:1502.02655*.

Software

- `dep-brown-cluster`: Available at `http://github.com/rug-compling/dep-brown-cluster`. This is a syntactic extension of the Brown et al. 1992 clustering algorithm.

- `hmm-reps`: Available at `http://github.com/rug-compling/hmm-reps`. Inducing word representations from latent-variable sequential and unlabeled or labeled tree models.
- `bimu`: Available at `http://github.com/rug-compling/bimu`. Inducing multi-sense word representations bilingually with discrete-state autoencoders.
- `align2tex`: Available at `http://github.com/rug-compling/align2tex`. Visualizing word alignments. It creates a LaTeX source from raw files produced by a word aligner.

PART I

Background

CHAPTER 2

Representing words

This thesis is about word representations. We study existing methods for inducing word representations and build on these to develop novel approaches by focusing on syntax and multilinguality. The word representations are an important theme in natural language processing (NLP) because they make it possible to think of words in terms of their semantic relatedness, which allows abstracting away from word occurrences in text and generalizing in a way that is beneficial in reducing lexical sparseness encountered in practical NLP applications.

The methods described in this thesis rest on the principle that the meaning of words can be effectively represented with mathematical objects such as categorical identifiers of semantic clusters to which words belong, or possibly with more complex and powerful vectorial representations. Word representations can be obtained or learned with various methods and using different definitions of input representation and context. They possess different properties in terms of representational capacity and, if they are induced from data, also in terms of computational complexity. Often, once we have successfully obtained the representations, we can apply them to test datasets in more than one way, depending on the properties of the underlying representation method. Here, we approach word representations

from within various frameworks. We examine the importance of word context definition primarily through the role of syntax in models of word representations; we build representationally richer models that are capable of handling polysemy; we induce models in a multilingual setting; and we study the ways of applying word representations and quantifying their quality in NLP tasks. In this chapter, we begin by discussing some basic principles such as capturing word meaning distributionally, why we need word representations, and how representation learning can be seen as a constituent part of supervised machine learning.

The semantic representation of language is a wide topic in NLP—cf. the coverage in Jurafsky and Martin (2008)—that ranges from word-level semantics, the composition of word representations into higher-order groups of phrases and sentences, to model-theoretic semantics and semantics of events and frames, to name just a few important directions. In this thesis, we only focus on word-level representations, and we will be asking the question how to construct word representations in such a way that they either approximate some human-encoded lexical semantic knowledge—this is the case when comparing against human-built ontologies and semantic similarity benchmarks—or, that they help improve a downstream NLP system, such as one for annotating words with parts of speech; a system for recognizing organization and location mentions; and an analyzer of semantic frames and its participants in the sentence. Sometimes, word representations are granted, and an approach might consist of simply using already existing hand-crafted semantic classes. In that case, we would like to solve the question of assigning words encountered in the text to some classes in the lexical resource. And since the domain of word-level semantics, like many domains in NLP, is subject to ambiguities, this process is often not trivial.

An obvious question arising is how to define word meaning and word similarity. We will adhere to the distributional hypothesis (Firth, 1957; Harris, 1954) according to which the meaning of a word is related to the contexts in which it occurs. So, two words that occur in similar textual con-

texts are said to have similar meaning. One of the important challenges we will be addressing in this thesis is what role the definition of context has in finally obtained word representations. Obviously, a single word can have different meanings depending on the context. We will find it useful to refer to such cases as different senses of a word: These are normally taken to be discrete representations of a word's meaning (Cruse, 1986). We can illustrate this with the time-worn example of *bank* that can be characterized with two radically different senses, one of a financial institution (example (2.1)) and one of a raised area of land (example (2.2)):

(2.1) The fear of many people is to one day be in a bank which is being robbed.

(2.2) A man was fishing on the opposite bank.

Here, the two senses are quite unrelated and would normally be classified as a case of homonymy. When the senses are related, we call the relationship polysemy. Such is the case for the *blood bank* and *sperm bank* sense (example (2.3)), which is related to the financial institution sense in that it is a repository for some entities (Jurafsky and Martin, 2008):

(2.3) He submitted a detailed plan for setting up a blood bank at the hospital.

For our purposes, the capability to distinguish between homonymy and polysemy in an automated way will not be important. Also, this distinction is in practice frequently not clear-cut, as shown for example in lexicography (Atkins and Rundell, 2008). Relations in which senses participate can be of different kinds. We will use the term semantic relatedness to mean any form of established relations, like synonymy, meronymy, antonymy, hypo-/hypernymy etc. (Turney and Pantel, 2010; Cruse, 1986).

The reason for adopting word representations is motivated by the generally accepted knowledge that in computational processing of the language, we need more than just bare word identities—we require additional information that would tell us what the words mean and how they relate to each other. Put differently, we need a way of bridging the gap from the

surface word level to the knowledge of the world that is needed to solve many higher-level natural language processing tasks.

2.1 Overcoming lexical sparseness

We now explain the problem of lexical sparseness, which is one of the central challenges in NLP. For this, we use the language modeling task as an illustration. In general, the goal of language modeling is to estimate the probabilities for all strings in the language. This is crucial for any task in which we need to identify words from a noisy input, or need to generate or rank word sequences, to name just a few applications. Let us take as example the four-grams having the pattern *known for his* $*$. As shown in Figure 2.1, based on a relatively small corpus of 1 million words, we find 32 candidates that could fill in the empty slot $*$. However, as much as 30 of them only occur once, which makes it difficult to construct reliable probability estimates. Another problem is that many candidate words have not been encountered in this small corpus at all. We can observe that this is the case by doubling the size of the corpus and repeating the counting (Figure 2.2). Several new words are now the possible candidates of the empty slot. Also, certain words from Figure 2.2 have now received more reliable counts, but the form of the distribution stays approximately the same in this case, and follows roughly the Zipfian distribution of words[1]. Note however, that even though we have increased the corpus size, the great majority of words are still hapax legomena, and we can still think of other legitimate word fillers that have not been observed yet. If we were to construct maximum-likelihood estimates for the words in the second list, we would assign some positive value to e.g. $p(\text{role}|\text{known}, \text{for}, \text{his})$, yet would need to give a probability of zero to $p(\text{movie}|\text{known}, \text{for}, \text{his})$, because "movie" was not encountered in the corpus. Regardless of how large the corpus, many words or word sequences will occur very infrequently or

[1] According to one of Zipf's laws (Zipf, 1949), a word's frequency is inversely proportional to its frequency rank.

will not be seen at all, which poses a challenge in NLP applications such as word-sense disambiguation (Ide and Véronis, 1998) and parsing (Bikel, 2004; van Noord, 2007). We will refer to the problem which we have just introduced with the above example as the *data* or *lexical sparseness* problem, that is, the fact that the data are obtained from a particular finite corpus, and that for many model parameters, an NLP system will have zero or very infrequent examples (Dagan et al., 1993; Jurafsky and Martin, 2008; Manning and Schütze, 1999).

One possibility to get around this problem could be to adopt one from the series of *smoothing* techniques that address the poor estimates of the maximum likelihood in language modeling. Another approach[2] would be to consider word generalizations in the form of word classes, following the *class-based* modeling of Brown et al. (1992), which was originally developed to deal precisely with sparsity in language modeling. Such a class-based approach works by representing words with hard classes, or clusters, obtained from distributional properties of the corpus; here, a word can belong to exactly one (semantic) class. We will discuss the details of a class-based approach to word-representation learning in Chapter 3 and Chapter 6. At this point, it is useful to note that by having access to semantic classes, we can use the counts of another word that is related and occurs in the same class as the word for which the relevant statistics are missing. For example, although we can not estimate $p(\text{movie}|\text{known}, \text{for}, \text{his})$ from the data (Figure 2.2), we can use $p(\text{film}|\text{known}, \text{for}, \text{his})$ as a reasonable substitute. Yet, to be able to generalize in this way, we need to have access to semantic classes or some other way which would let us relate words that are semantically similar, like "movie" and "film" above.

[2]In practice, a common way of alleviating the sparsity is also to use abstraction in the form of pre-processing (e.g. down-casing, lemmatization, substituting numbers and one-time word occurrences with special symbols) and parts of speech (Smith, 2011). The downside of the first is that it is ad hoc and might need to be altered depending on the dataset or the language used. The downside of the second is that POS categories might be too coarse and would excessively abstract away from the word level.

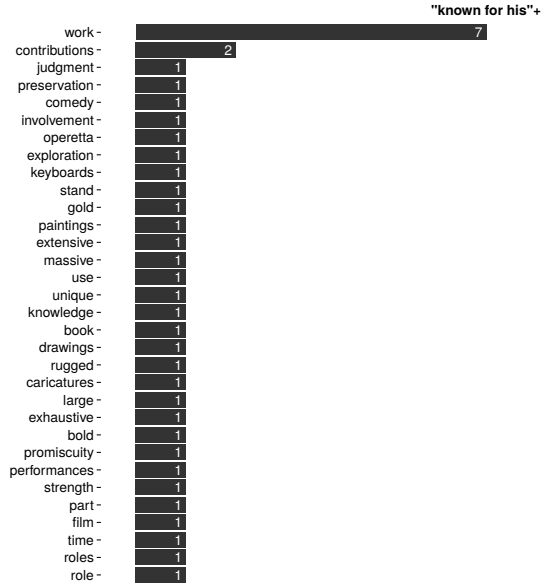


Figure 2.1: Frequency distribution of words following the sequence *known for his*. The statistics gathered from the first *1 million* words of Wikipedia (Shaoul and Westbury, 2010).

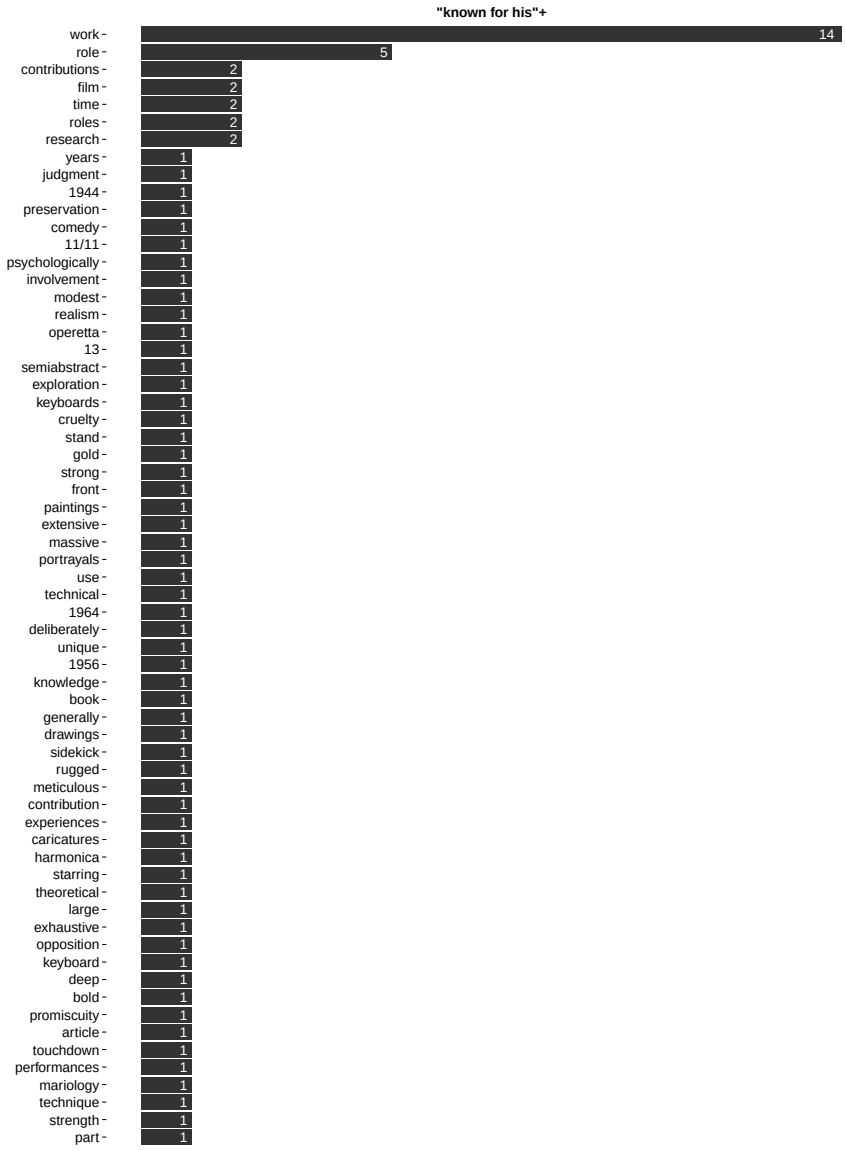


Figure 2.2: Frequency distribution of words following the sequence *known for his*. The statistics gathered from the first *2 million* words of Wikipedia (Shaoul and Westbury, 2010).

2.2 Applicability of word representations

We will now discuss how the word representations are used and how their quality is evaluated. The learning of word representations is not an end in itself, and most commonly we would like the learned representations to contribute to a concrete NLP application. To measure the quality and compare the word representations, one possibility is then to use the NLP task we are ultimately interested in, and measure the effect on the performance. This type of evaluation is called *extrinsic*. The evaluation can also be *intrinsic*[3], meaning that testing is performed independently of a given task. For this purpose, semantic similarity benchmarks or human-built lexical inventories and ontologies are often used. Overall, the advantage of intrinsic evaluation lies in its ability to provide a general account of the quality of representations that is not task specific and is cheap to obtain; it can also help separate out the strong and the weak aspects of word representations, and describe specific meaning properties captured when using dedicated benchmarks. Additionally, intrinsic evaluation in word-representation learning often goes hand in hand with qualitative evaluation, in which an attempt is made to improve the representation model introspection and enhance human interpretability. For example, in the case of vectorial representations, similarities can be measured and words can be related graphically in a two-dimensional space. The disadvantage of intrinsic evaluation is that the outcome might not necessarily correspond to the performance one would obtain in an extrinsic task. Another related reservation is that intrinsic evaluation often depends on human judgments and human-built categories, but optimizing for them might not necessarily lead to improvements in an end application.

We now try to generalize the application of word representations in downstream NLP tasks such as POS tagging, named-entity recognition, parsing, sentiment analysis, question answering etc. In this context, we can think of word-representation learning as feature learning whose suc-

[3]The *in vivo*/*in vitro* terminology is sometimes also encountered in the literature.

cess we try to evaluate on a particular task. A supervised machine learning predictor can use the learned features as input for the learning of its own parameters (Bengio et al., 2013). It might also include other, manually-coded features to capture salient effects for a particular task (for example, a salient feature in named-entity recognition is word capitalization). Still, especially in neural network-based approaches, it is widespread to use word representations as initializers at the input layer without any other hand-engineered features (Goodfellow et al., 2016). One property of representation learning that distinguishes it from other machine learning tasks like classification is the lack of a clear training objective, which is far-removed from the ultimate objective to learn a classifier of some sort (Bengio et al., 2013). Larochelle et al. (2007) and Bengio et al. (2013)[4] emphasize the fact that many classification tasks are made complex by an interaction of factors arising from complex data, like encountered in image or natural language processing; the role of representations in these tasks is therefore also *to disentangle the factors of variation*. In NLP, the factors of variation behind the produced texts in the language can be variables such as topics, domains, author characteristics that include sociolinguistic variables like age, gender and place, as well as more pragmatic variables like space constraints when writing texts.

2.3 Formulating the representation learning problem

Although we mainly work in this thesis with word representations as some entities to be *learned*, we keep a broad perspective and include also those categories that were *hand crafted*, as long as they fulfill the goal of providing a generalization that will be useful in downstream tasks. One representative of the latter category are the categories and relationships from human-constructed lexical inventories, e.g. wordnets. Despite the fact that the categories are given, one still needs to find the right level of generaliza-

[4]Note that their view of representation learning is much narrower than ours, and encompasses only the representations rooted in probabilistic graphical models and neural networks.

tion for these representations and design a mapping between the tokens encountered in the corpus and the existing semantic categories. We put such an approach in practice in Chapter 5.

We now turn to those representations that are learned from data. We have used the word *learning* already at several places in the thesis. We borrow its definition from Mitchell (1997):

> A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

By learning of word representations, we therefore simply mean the optimization of parameters of a word representation model in a way that leads—based on training corpora—to increased performance on an intrinsic or extrinsic task.

Many approaches that use word representations in downstream NLP tasks can be often characterized as instances of *semi-supervised learning*. This is the case whenever the task can be decomposed in unsupervised learning of word representations (from unlabeled data) followed by standard supervised learning (Huang et al., 2014). The used representations can be task-independent in the sense that they can be applied to any task once learned. However, it is true that their success on these tasks can largely depend on the domains of the texts on which they are trained, and on the domains on which they are tested.

Following Huang et al. (2014), we can now specify the representation learning problem a bit more formally. Recall that in a traditional supervised machine learning task, we want to optimize a hypothesis on labeled training data and use it to make predictions on previously unseen test data. By optimization of a hypothesis we mean the learning of model parameters that is itself based on features, or data representation. For a certain learning problem, we denote the instance space as X and the distribution or the domain from which X is drawn as D. Let Z be the space of pos-

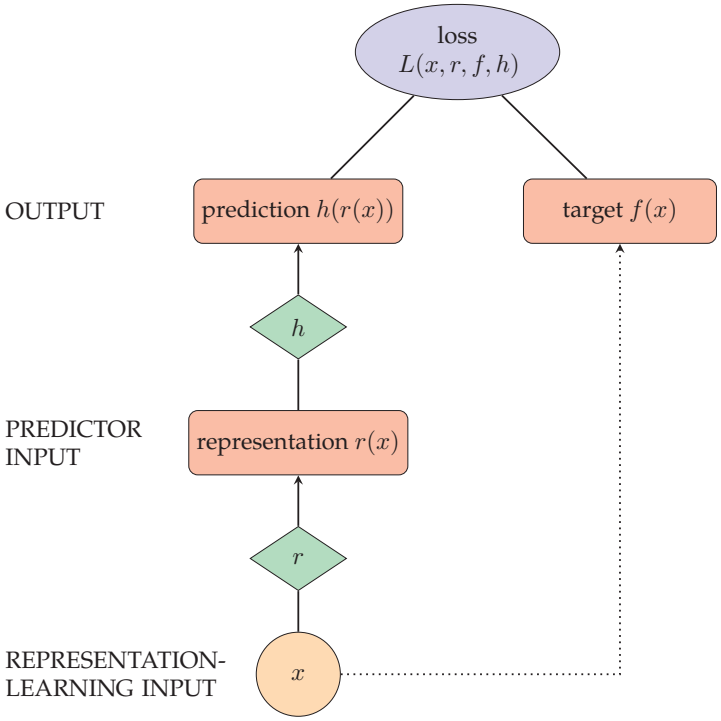


Figure 2.3: Representation learning as part of supervised machine learning.

sible labels for an instance, and let $f : X \rightarrow Z$ be a target function that assigns each instance its corresponding target (label). In POS tagging, for example, the instance set X is the set of all sentences and Z is the space of POS sequences (including POS labels like NN, for noun) that get assigned to the sentences by the target function f. Each position in the sentence is normally represented as a vector whose values correspond to values of features. We also assume the existence of a *representation* function $r : X \rightarrow Y$, which is a mapping from instances to the feature space Y, which is typically high dimensional. We refer to the dimensions of Y as *features*. Given the training examples, the overarching goal is to select a hypothesis h from

the space of possible hypotheses H, so that some loss function which measures the cost of the mismatch between the target function and the hypothesis is minimized. Here, we include in this formulation also the learning of representations:

$$r^*, h^* = \arg\min_{r \in R, h \in H} \mathbb{E}_{x \sim D(X)} L(x, r, f, h), \tag{2.1}$$

where r^* and h^* are the best representation and hypothesis, respectively, identified by the optimization step that minimizes the expectation $\mathbb{E}$ of the loss $L(x, r, f, h)$ for all instances x drawn from $D(X)$. Since the predictions depend on the representation function, namely $h(r(x))$, a good representation is the one that leads to better predictions and consequently to a smaller expected loss[5]. At test time, r^* and h^* can be used to obtain predictions on previously unseen data. For example, r^* would establish a representation in the feature space for a sentence that we would like to tag, and h^* would assign the POS labels to each of the words in the sentence. A diagram of how supervised training relates to representation learning is shown in Figure 2.3. In training, we are comparing by means of a loss function (shown as the purple oval-shaped node) two different outputs: the output of the machine learning predictor ("prediction $h(r(x))$") and the output of the target function that represents the gold-annotated labels. The underlying input x is the same in both cases. For the machine-learning part on the left, the prediction is a result of applying a hypothesis h (green diamond-shaped node) to a learned transformation r of the input data x to the feature space.

[5]In the context of POS tagging, the loss could, for example, simply indicate the number of words tagged differently in the target $f(x)$ and the hypothesis $h(r(x))$.

CHAPTER 3

Selected approaches to word representations

In this chapter, we describe different frameworks for obtaining word representations. We will not attempt to cover comprehensively all existing ways of representing words, but we will focus on those research directions that are relevant for the subsequent chapters of the thesis. We thus give a high-level overview of *concept-based semantic representations*, *word clustering* and *latent-variable models*, and *word embeddings*, but we omit, for example, the *distributional semantic models* as described in Turney and Pantel (2010), Sahlgren (2006) and Šuster (2012), which are an important research direction that in its narrow definition includes high-dimensional vector space models and dimensionality reduction techniques that can be applied to them (Deerwester et al., 1990; Lund and Burgess, 1996; Landauer et al., 2007).

As we have mentioned in the previous chapter, the first framework, which we describe in section 3.1, is special in that it is the only one in which we assume that the word classes are given, or put differently, they are not learned distributionally. In the frameworks of clustering and latent variable models, and that of word embeddings, the word representations as

well as optionally the underlying word sense assignment are obtained or estimated from data. Since the true outputs are not given for any training examples, we call this setting *unsupervised*. For some unsupervised applications, the outputs could be obtained—perhaps at great expense—to create an annotated dataset and then evaluate the learned predictor. One example of such an application is unsupervised grammar induction, in which the goal is to infer grammatical structure for sentences without the use of a treebank (Smith, 2011), but the evaluation can be based on a treebank. However, in other cases like word class induction, "correct" solutions might not be knowable by humans, and the evaluation is often conducted on some larger extrinsic task.

In the next sections, we present the different frameworks in the order that they follow throughout the thesis. These are not intended to serve as a review of all the work done in each field, as we review the works related to ours in individual chapters. They are meant rather as an elementary introduction on which we build in the following chapters.

3.1 Concept-based semantic classes

Concept-based semantic classes, shortly *concept classes*, are word classes obtained from manually-constructed lexical inventories, ontologies, dictionaries, or other machine-readable linguistic resources. Examples of such resources include wordnets[1] for numerous languages, knowledge bases like Freebase[2], frame-semantic resource Framenet[3] and Roget's thesaurus (Roget, 1911).

A concept class is a grouping of word types sharing the same concept. Taking wordnets as an example, a *synset* is a concept class that groups synonymous word-POS pairings in a single set, and records relations between the sets or its members, thus forming a hierarchy. It is possible that a sin-

[1] We do not capitalize the word when we refer to the resource in a generic, language-independent way.

[2] `www.freebase.com`

[3] `http://framenet.icsi.berkeley.edu`

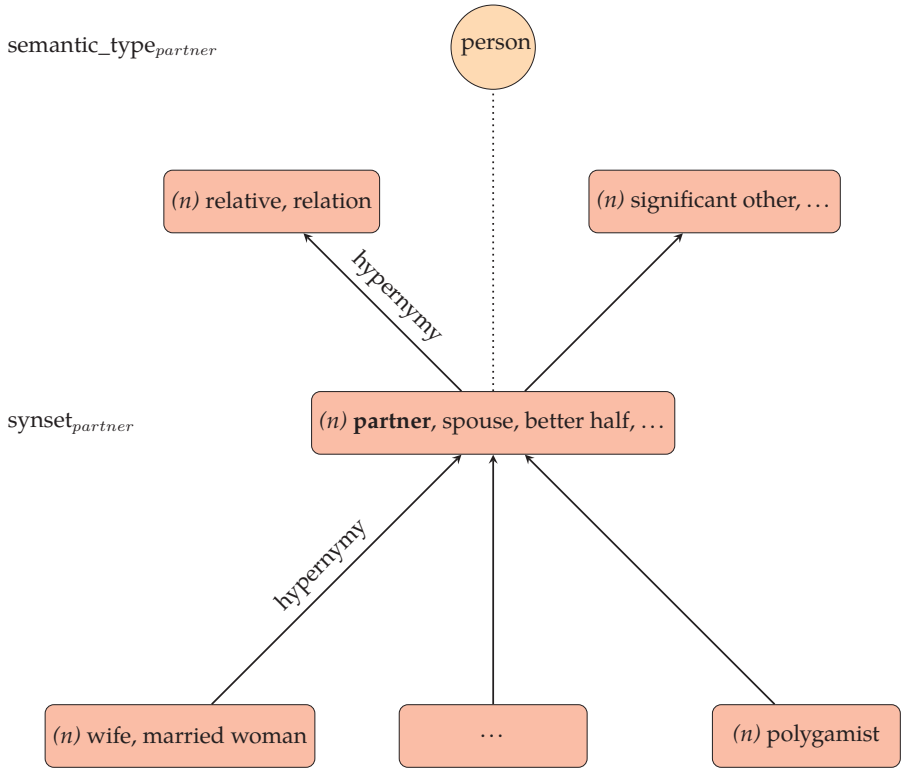


Figure 3.1: Example of concept classes and hypernymy in English WordNet for one of the senses of the noun *partner*.

gle word (more precisely, word-POS pairing) simultaneously belongs to several concept classes. The usage of concept classes is closely connected to the issue of lexical ambiguity, which is the object of research of word sense disambiguation, i.e. establishing the mapping between the observations in a corpus and the concept class entries in a lexical resource (Ide and Véronis, 1998). In feature engineering for a ML predictor, the use of concept classes is widespread; and it is also common to use related classes such as the ones obtained through hypo-/hypernymy relations (Faruqui

and Dyer, 2015). We give an illustration of a wordnet hierarchy based on a concrete noun in Figure 3.1. Wordnets like WordNet for English (Miller, 1995; Fellbaum, 1998) and Cornetto for Dutch (Vossen et al., 2013) also include semantic types (sometimes also called semantic fields or lexicographer classes), which can be used to incorporate more coarse-grained semantic categories in an NLP system. The associated task of mapping word occurrences to semantic types is usually referred to as *supersense tagging* (Ciaramita and Altun, 2006). Examples of semantic types are the "animal" type for nouns and "emotion" for grouping verbs describing feelings. Around 20 such types exist in Cornetto, and 26 in English WordNet.

Concept classes are widely used in feature engineering for supervised ML tasks, examples being syntactic parsing (Agirre et al., 2008) and frame-semantic parsing (Das et al., 2014). They can also serve as constraints in distributional learning of word representations (Faruqui et al., 2015; Xu et al., 2014; Yu and Dredze, 2014). Research has been conducted also in the opposite direction, so that distributional methods are used to augment linguistic resources like wordnets with new synonyms and relations (Hearst and Schütze, 1996; Harabagiu et al., 1999; Widdows, 2003).

Our concrete use of these resources in this thesis amounts to adopting Cornetto, a wordnet for Dutch (further details in section 5.2.3), and the concept classes extracted from it for obtaining word representations that help deal with lexical sparsity in syntactic parsing of Dutch. Furthermore, we also use both Cornetto and the English WordNet when evaluating the quality of induced word clusters in Chapter 6. In that case, the success of inducing word clusters is measured as the degree to which the clusters match the reference concept classes created by humans.

3.2 Clustering and probabilistic latent-variable models

Word clustering describes a family of approaches for grouping distributionally similar words together. As this is an unsupervised setting, the obtained groups do not carry names. A common scenario is to simply use the

identifiers of clusters to which words belong as features in a ML task. The literature normally distinguishes between clustering algorithms depending on the type of induced structure (Duda et al., 1973). If the structure is flat, the algorithm is called *partitional*. If it follows a structure in such a way that we can relate one cluster to another, and so that some clusters are specializations of more general clusters, the algorithm is categorized as *hierarchical*. The hierarchical approaches can be further divided in *agglomerative* and *divisive* approaches, depending on whether the structure is obtained in a bottom-up manner, i.e. starting with individual words and then merging them to form clusters (agglomerative), or in a top-down manner, in which all words initially share a single cluster, but would be subdivided (hence, divisive) as the training proceeds.

Brown clustering (Brown et al., 1992), the method used in this thesis, is a well-known representative of a hierarchical agglomerative method that relies on counts of bigrams obtained from a corpus. The induced structure is a tree, and the word clusters are located at the leaves. The leaves closer to each other are more similar than those further away. The tree structure also conveniently allows obtaining coarser clusters by choosing nodes higher in the tree to achieve more general representations. We describe the algorithm in detail in Chapter 6. Some example clusters are shown in Figure 3.2, forming part of a subtree from a larger tree containing 1000 word clusters induced from Dutch tweets[4].

One of the most widely used and studied unsupervised algorithms that performs clustering of any data instances, not just words, is the *k-means* algorithm. The idea is to represent data characteristics with feature vectors and to induce clusters of similar data points that partition the data. This is done by representing clusters with their prototypical vectors, called *centroids*. We begin with a guess of cluster centroids, and assign the feature vectors to their closest centroids. We then move the centroids to the average of data points in a cluster. This procedure is performed iteratively until the cluster centroids stop moving. In k-means, the number of partitions k

[4] `http://www.let.rug.nl/gosse/Ngrams/brown.html`

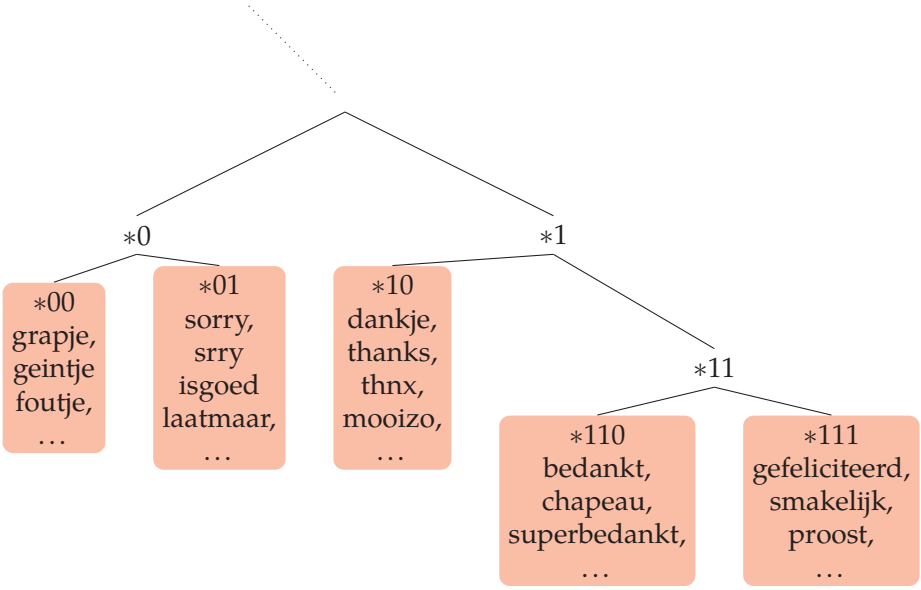


Figure 3.2: An excerpt from a Brown clustering tree induced from 6 million Dutch tweets. English translations are: *grapje, geintje→ joke, foutje→ mistake, isgoed→ it's OK, laatmaar→ nevermind, dankje→ thanks, mooizo→ nice, bedankt→ thank you, chapeau→ hats off, gefeliciteerd→ congratulations, smakelijk→ bon appetit, proost→ cheers*.

needs to be specified by the user. K-means can be seen as a general clustering algorithm that leaves the feature construction, i.e. the exact specification of what each feature dimension represents, to the user. For example, in word clustering, different context types would lead to different feature vectors used as input to clustering. We refer the reader to Manning et al. (2008) for a detailed overview of this clustering algorithm and its applications.

Unlike in Brown clustering in which the cluster membership of words is defined by a Boolean function (often referred to as *hard* clustering), it is possible to learn a cluster membership function that is probabilistic, such

as in Pereira et al. (1993) and Rooth et al. (1999), for example. In their work, the goal is to hierarchically group nouns according to their conditional verb distributions. In this way, they represent cluster membership as a probability distribution over clusters (or senses), $p(c|w)$. Such a method can be described as *soft* clustering.

More generally, many probabilistic models associate latent variables with distributionally induced classes of interest and associate each word (or another observed variable) with a probability distribution over those classes. In probabilistic modeling, the latent variables represent random variables whose values are not observed in the input data. In the context of word representations, a latent variable can be introduced to capture the sense of a word, either by taking a single value or a distribution over the values to represent the word sense(s). A case in point is a Hidden Markov model, which is based on two assumptions: First, the probability of a word appearing depends only on its own semantic class, which is not observed in the unsupervised case; second, the probability of a semantic class appearing is dependent only on the previous semantic class. The model can be adapted, though, so that the relationship between the classes is not temporal or sequential, but for example structural. We will introduce methods belonging to this family of models with varying definitions of class relationship in Chapter 7.

To obtain word representations from an induced latent-variable model, and specifically in a Hidden Markov model, inference is required. This finds the most probables states given the observations at test time. In Figure 3.3, the most probable states representing semantic classes are the darkest-colored nodes in each word vector. In this illustration, the state size is only 5, although in reality, it is usually in the order of hundreds of states. Given the sequence "the significant other", we could use the indices of the most probable states (class identifiers) "1-2-0" as discrete representations for the words in the sequence. The flexibility of this framework lies also in a wide array of options for obtaining word representations. For example, we could also use the entire vectors as representations to increase

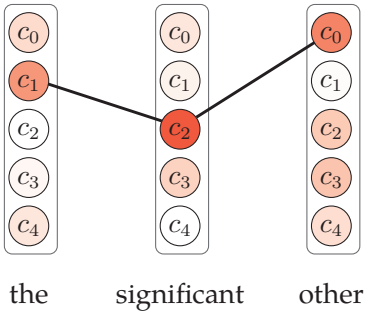


Figure 3.3: Illustration of vectorial representations inferred from a Hidden Markov model, with an optional selection of the most probable states.

the representational capacity. We describe these and other possibilities in Chapter 7.

The number of states in such a model, set to five in the above example, might be either a parameter to the model that needs to be set in advance or it can also be discovered through learning using a mechanism controlling the creation of new classes depending on the environment encountered at each training instance[5]. While increasing the number of classes in a model is likely to increase the capability to capture fine-grained distinctions between some items, it is also true that this finer granularity can come at a cost of reduced generalization.

3.3 (Neural) word embeddings

Word embeddings bear some similarity to the latent variable models, especially when these are used in conjunction with techniques for obtaining *vectorial* representations. For example, word embeddings are also real-valued vectors and they are typically low-dimensional (up to hundreds of dimen-

[5]One subset of approaches falling in the second category contains Bayesian nonparametric models, see for example (Séaghdha and Korhonen, 2014).

sions, rarely more than one thousand).[6] However, they usually do not have a probabilistic motivation or interpretation, and originate from the research on neural networks. In addition, the standard embedding models do not provide distinct representations for a word type occurring in different contexts. One advantage of word embeddings is that they are fast to train, thus scaling well to datasets with billion words and more. In comparison to distributional semantic models in which high-dimensional, sparse representations are created (and no dimensionality reduction is applied), the word embeddings are dense representations, meaning that most of the values are not zero.

Word embeddings are also known as distributed representations, in which *distributed* refers to a "many-to-many" relationship between a concept and the vectorial representation. Namely, each concept is represented by a so-called pattern of activity distributed over many computing elements, and each computing element participates in representing different concepts (Rumelhart et al., 1986). In this way, two similar concepts would display a similar pattern of activity, unlike in a scheme like *one-hot encoding* in which only one element in the vector is activated to represent a word identity, and in which a particular word is equally similar or dissimilar to all other words.

The use of word embeddings for representing words have been explored in the context of language modeling by Bengio et al. (2003), and later perhaps most notably by Mnih and Hinton (2007), Collobert and Weston (2008) and Mikolov et al. (2013a). The embeddings are learned in a network by a prediction task, in which one word w_p is predicted based on another word w_i or a combination of words, e.g. $p(w_p|w_i)$. The embeddings actually just represent the weights needed to be tuned in order to perform the prediction task well. Usually, separate embeddings are trained for the input words and the predicted words. The intuition behind the capability of embeddings to capture semantic similarity between words lies in an

[6] This characteristic is beneficial when including the embeddings as features in a ML task, since it allows for faster training.

observation that similar words occur in similar contexts, thus two similar input words will need to successfully predict the words they occur with. In the Skip-Gram method of Mikolov et al. (2013a), the task of computing the probability $p(w_p|w_i)$ boils down to computing the dot product between the input word's embedding and a context word's embedding[7], and turning it into a probability by applying a softmax function. This operation is then repeated for every input word in the corpus and the embeddings are updated using gradient-based techniques. Since the representation of the input word needs to become more like the representation of the predicted word, and since many predicted words for two similar input words are the same, the representations of the input words must also become alike. We discuss the details of training this model as well as other, more complex models of word embeddings in Chapter 8.

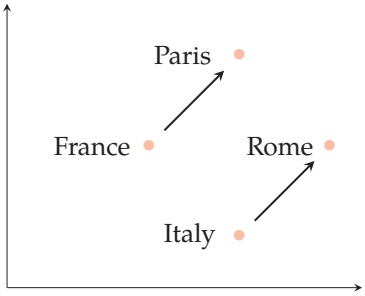


Figure 3.4: Analogical relationship as one of the regularities observed in word embeddings.

The embeddings have been shown to exhibit some interesting properties, like that of analogy, which reveals that semantic relations can appear as linear relationships in the vector space. For example, if we plot the vectors of "Paris", "France", "Rome" and "Italy" (Figure 3.4) by projecting

[7] An interesting relationship between the Skip-Gram and the count-based pointwise mutual information (PMI) has been found by Levy and Goldberg (2014c), according to which the product of the input and context embedding matrices corresponds to a PMI matrix with some offset.

them on a two-dimensional plane, we can use it to obtain approximate answers to questions like "Paris is to France as ____ is to Italy". This has been shown to work for several types of named entities and beyond.

Once word embeddings are trained, a common application is to initialize embedding layers in neural network predictors, which are intended to solve some concrete NLP task, for example dependency parsing (Chen and Manning, 2014). Such embeddings can be either kept fixed during training or fine-tuned to the task at hand. In the deep learning community, the step involving the embedding training is sometimes called *unsupervised pretraining* (Goodfellow et al., 2016; Collobert and Weston, 2008). Although the word embeddings can be learned from scratch when forming a part of a neural model optimized for a particular task, the amount of (gold) data is normally limited, and the embeddings learned this way would be constrained to the specific task only.

The embeddings are a vigorous line of research in NLP with a vast amount of work done in many directions. Some of these include alternative model architectures (Pennington et al., 2014); effect of context type (Levy and Goldberg, 2014a); comparison to count-based distributional semantic models (Baroni et al., 2014); the treatment of larger units like phrases, sentences and documents (Mikolov et al., 2013b; Le and Mikolov, 2014; Kiros et al., 2015); the extensions to account for polysemy (Neelakantan et al., 2014); the study of evaluation types (Schnabel et al., 2015); and multilingual learning of word embeddings (Klementiev et al., 2012).

CHAPTER 4

Applications for word representations

In Chapter 2, section 2.2, we have touched upon the applicability of word representations and made a broad distinction between intrinsic and extrinsic applications. We have also introduced a conceptual schema, in which we position representation learning as a part of (semi-)supervised machine learning. In the previous chapter, we have presented an overview of different approaches to obtaining or inducing word representations. We now describe each concrete application of word representations in turn—limiting ourselves only to the tasks actually used throughout the thesis—to provide the high-level background information important for the content that we present in the following chapters. The different types of applications studied in this thesis are shown in Figure 4.1. We evaluate the majority of the methods on more than a single task; also, we carry out two evaluation tasks with more than one method.

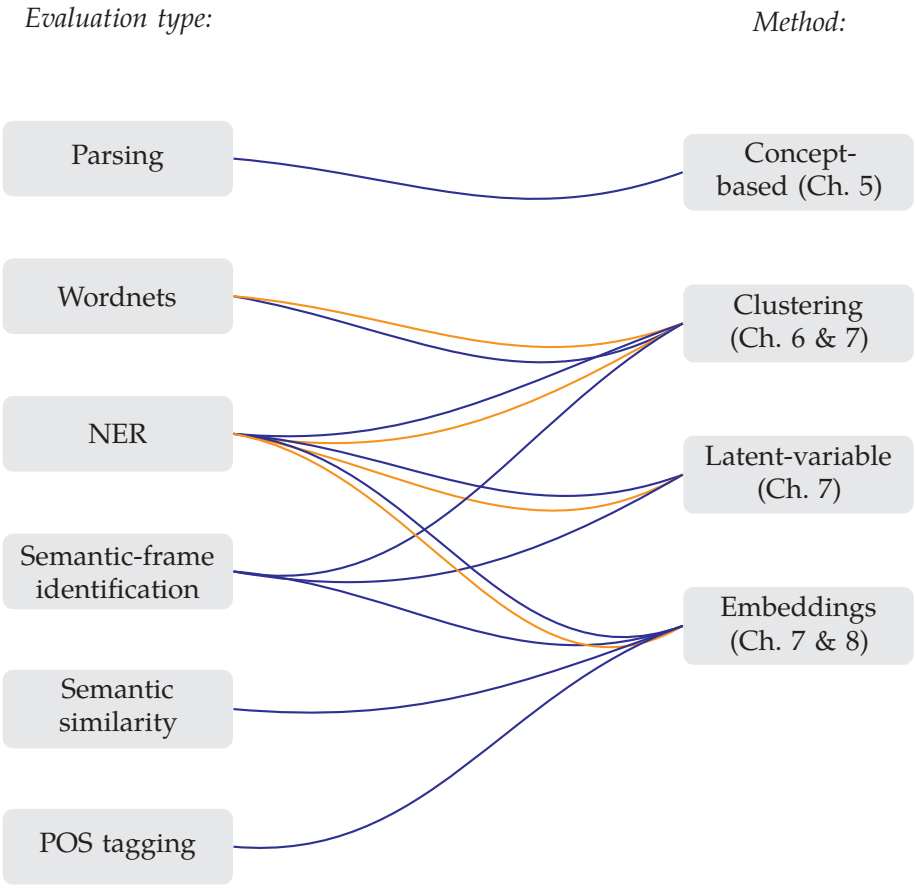


Figure 4.1: Types of evaluation carried out in this thesis, matched with the word representation method used. The information in parentheses refers to the chapter in which the results are discussed. The lines in blue indicate that the evaluation is carried out in English; the orange indicates Dutch. For "Embeddings", only a baseline method is evaluated in all four applications to which the links exits; the newly developed embedding models from Chapter 8 are studied empirically only with the "Semantic similarity" and "POS tagging" methods.

4.1 Intrinsic applications

4.1.1 Semantic similarity

One possibility for evaluating word representations is to adopt a semantic benchmark in which the similarities of word pairs were estimated by human raters. Many such benchmarking datasets exist, especially for English (the ones used in this thesis are listed in section 8.6.1); some of them are specialized in measuring a particular kind of semantic properties, some are intended to cover particular sets of words according to, for example, frequency or domain criteria. The evaluation procedure for word representations in the context of semantic similarity benchmarks consists of measuring how well the automatically obtained similarity scores on the word pairs from the benchmark match the ratings produced by humans. As the automatic similarity scores are normally real-valued, and the (aggregated) human ratings are either ordinal or real-valued, some kind of correlation analysis can be performed, indicating the extent of linear or non-linear relationship between the automatic and human ratings. In this setting, a word representation method is deemed better than some baseline method if it better correlates with human judgments.

A well-known semantic similarity benchmark is WS353 (Finkelstein et al., 2001), which we mention here to illustrate the nature of the task and the presentation of the benchmark format. In it, 353 word pairs are given one per line together with their similarity scores in the range between 0 and 10, averaged over several human ratings:

(4.1)

word 1	word 2	avg	rat. 1	rat. 2	rat. 3	…
professor	cucumber	0.31	1	0	0	…
money	cash	9.15	10	9.5	9.5	…
marathon	sprint	7.47	7	9	7	…
profit	loss	7.63	8	9.5	9.5	…

In this dataset, we also find antonymous pairs, such as “profit–loss”; the human raters were asked to judge such pairs as similar provided they be-

long to the same domain.

The problematic aspect of the benchmarks based solely on word pairs is that no context is given that would allow the raters to compare word senses rather than possibly polysemous word types. This concern has been addressed, for example, in the Stanford's Contextual Word Similarities dataset, known shortly as SCWS (Huang et al., 2012), which includes around two thousand word pairs, each word appearing in the context of a specific sentence (additionally, some sentences to the either side of the target sentence are included). The set of words in SCWS was chosen based on Wordnet, and for each word pair in context—mostly noun–noun, but also verb–verb, noun–verb, adjective–adjective, and some others—ten human judgments were obtained on a scale from 1 to 10. An example data instance from SCWS is shown here:

(4.2)
[...] involved the *murder* of almost 7,000 priests [...]
[...] murder and *manslaughter* . This sentiment has been [...]
Average rating: 7.6
Individual ratings: 8, 8, 10, 10, 9, 5, 4, 10, 5, 7

The words compared are shown in italics. For brevity, we omit here some sentential context to the either side. In this particular instance from the benchmark, the target words were assigned a relatively high average similarity score. The advantage of contextualizing word pairs is relevant both in the rating phase to obtain more accurate human judgments and in automatic scoring in which the context can be exploited by the word representation methods that can perform context-sensitive inference, and thus come up with a different representation for a word depending on its context.

Regardless of the benchmark or the word representation method used, in order to compare automatically obtained scores with human judgments, we need a way of obtaining the automatic scores in the first place, i.e. we need a way of measuring the similarity between the representation for each word in the pair. For vectorial representations, this is typically done by using a similarity metric such as the *cosine*, which calculates the similarity

as a dot product over unit-length vectors, and outputs a score in the range of 0 (perfect dissimilarity) and 1 (perfect similarity).

A different approach to the intrinsic evaluation of word representations than the one described so far is *supersense similarity*, in which the ground truth is still available in the form of human judgments, but it is encoded based on human annotated corpora. In the method we use, the human annotations of (super)senses in a corpus are encoded as a feature matrix consisting of "linguistic" vectors, which then serves as a reference to which we compare the induced word representations (Tsvetkov et al., 2015). The comparison works by aligning dimensions of a vectorial word representation to dimensions of linguistic vectors by maximizing the cumulative correlation or the cosine score of the aligned dimensions. We use this method to quantify the quality of induced word embeddings for English in Chapter 8.

4.1.2 Wordnet-based similarity

The wordnet-based similarity measures are particularly well suited for obtaining information about similarity of discrete (categorical) word representations such as word clusters. These measures rely on the wordnet structure to produce a score—usually between 0 and 1—that quantifies the degree to which two concepts (synsets) are similar (Pedersen, 2010).

Taking clustering as an example, the gist of evaluation for word representations is to obtain a score of semantic consistency and similarity between the words that share the same cluster after the clusters have been induced. In other words, we are interested in how successfully a representation-inducing method groups similar words together. More precisely, a course of action might consist of the following steps:

(4.3)
1. choose a word w_i,
2. look up its discrete representation c_i,
3. for every $w_j, w_j \neq w_i$ also represented by c_i, calculate $sim(syn(w_i), syn(w_j))$.

In step 1, we begin by choosing a word, an action that can be guided by a pre-built list of target words. This is advantageous compared to evaluating all possible pairs of words in a clustering firstly because it is more efficient, and secondly because it ensures greater comparability—the set of words that are evaluated is the same regardless of the clustering method used. Since we use wordnet-based evaluation only with discrete representations in this thesis, the step 2 is trivial as we only need to look up the cluster in which the word type occurs. In step 3, we only consider the word pairs that share the same representation c_i. We therefore measure the most basic type of semantic similarity provided by clustering methods, which is the similarity of words sharing the same cluster. We could also extend the treatment to consider words within all clusters that are somehow related. If the clusters are organized in a hierarchy (like the one shown in Figure 3.2) with those clusters that are closer in meaning located closer in the hierarchy, it is possible to include in the similarity calculation also the neighboring clusters. The similarity function sim is intentionally underspecified in this step: It leaves a lot of room for different engineering solutions with respect to the exact calculation of the wordnet similarity. In fact, the problem concerns two choices to be made: the selection of the similarity metric and the mapping or the relationship between the word type and its synset(s). The purpose of a similarity metric is to produce a score based on the distance of two nodes in the wordnet hierarchy, optionally accounting for the location of the nodes in wordnet (higher or lower in the hierarchy) by introducing a specificity metric for nodes. One such metric is the well-known information content, proposed by Resnik (1995). Furthermore, the mapping function can also be implemented in different ways. For word types belonging to more than one synset, one straightforward approach is to calculate the similarity for all possible pairs, and then either take the highest scoring combination or the average over all possible combinations. We discuss these choices in more detail in section 6.4.

U.N.	official	Rolf	Ekeus	heads	for	Washington	.
I-ORG	O	B-PER	I-PER	O	O	I-LOC	O

Figure 4.2: Annotation of an English sentence with named entities. Adapted from the CoNLL-2003 dataset.

4.2 Extrinsic applications

4.2.1 Named-entity recognition

Named-entity recognition (NER) is a task of recognizing the occurrences of proper names in a text, including their spans, and of labeling them with entity types they represent. The exact definition of what constitutes an entity is usually task- and domain-specific.[1] It is a common practice in newswire texts to treat as named entities all names of persons, companies and locations, as well as sometimes the expressions referring to times and numbers. In other domains, one might be interested in recognizing the names of genes or football clubs, for example.

NER is most commonly treated as a sequence prediction task (Ratinov and Roth, 2009), using methods like Hidden Markov models, Conditional random fields and neural sequence models. Often, gazetteers—the extensive lists of named entities—are used to increase the performance (Krishnan and Manning, 2006).

The named entities in a text may often consist of more than one word. For this reason, the BIO scheme is usually adopted that encodes a word as either B(eginning), I(nside) or O(utside) of the entity. Generic NER benchmarks that are oriented towards annotation of news-like texts typically include four entity types—PER (persons), ORG (organizations), LOC (locations) and MISC (miscellaneous)[2]. The identification of entities and their

[1] It is beyond the scope of this work to discuss possible ways of determining which words and word sequences represent named entities. We work with already annotated corpora, for which it is reasonable to assume that some annotation guidelines were followed during the annotation process.

[2] MISC includes, among others, names of religions, languages, works of art and

word spans according to the BIO scheme prepends the entity label with either B or I, and simply use O for all words that are not named entities. An English example is given in Figure 4.2. The goal of a NER system for this sentence is to correctly recognize which words represent entities and which do not, as well as to assign the entities the right labels. This process is generally ambiguous—"Washington" can refer to either a location or a person, and in a more elaborated annotation scheme perhaps also to a political entity or a sports club. There is another type of ambiguity that goes in the opposite direction, i.e. from the named mention to different entities of the same type. For example, "Washington" as a person can refer to different people. This, however, is usually not seen as part of NER, but as a reference resolution problem (Jurafsky and Martin, 2008).

For Dutch and English, CoNLL shared tasks of 2002 and 2003 have provided the annotated datasets, now standardly used in the development of NER systems for general texts. These benchmarks include roughly 300,000 words for each language. Out of these, 17% represent named entities in English, and 9% in Dutch.

The particularity of NER in comparison to other sequence-labeling tasks is that the named entities are mostly nouns and noun phrases, many of which occur only rarely in the text. In an attempt to design a successful semi-supervised approach, word representations can be leveraged to reduce lexical sparseness by abstracting away from the given nouns and words surrounding them. Furthermore, context-dependent representations may succeed in capturing some ambiguities by actually inducing different representations for the same words occurring as different named entities in the text (for example, "Washington" as a location name and as a last name).

4.2.2 Part-of-speech tagging

Part-of-speech (POS) tagging is a process of assigning parts-of-speech to words, i.e. the categories based on syntactic and morphological function

sports-related names.

of words: Similarly functioning words with respect to their neighborhood or with respect to their morphological properties are grouped into classes. The most prototypical categories are nouns, verbs, adjectives and adverbs. Unlike in word-representation learning, the focus in POS tagging is not on semantic coherence within the categories, although it can come about in some situations.[3] The number of classes in English is typically lower than e.g. in word clustering whose primary goal is grouping words based on semantic coherence.[4] POS tagging is one of the most explored NLP problems, which have been tackled with various rule-based and statistical methods. The sequence-prediction models that can be applied to NER can also be used for POS tagging, although the two problems are different in that NER can be seen as a two-step process of first identifying the entities and then classifying them, whereas in POS tagging, every word bears a part of speech.

The use of POS in NLP applications is widespread and includes tasks such as syntactic parsing, information retrieval, authorship attribution, speech processing and others. Furthermore, POS annotation is also commonly used in corpus linguistic research (Jurafsky and Martin, 2008). The POS taggers for English newswire texts are most often trained and evaluated on the Wall Street Journal (WSJ) part of the Penn Treebank, which includes in total around 45,000 sentences, or 1.1 million words annotated with parts-of-speech.

The example in Figure 4.3 shows a gold standard annotation of an English sentence taken from the Penn Treebank. Several tags are used: CC for a coordinating conjunction, DT and NN for a determiner and a noun, VBZ for a verb in 3rd person singular, JJR for a comparative adjective, IN for prepositions and subordinate conjunctions, and PRP for personal pro-

[3]For example, a basic distinction in meaning can carry over to the distinction between homographs with different parts-of-speech. In general, however, many sense distinctions may exist inside the homograph, which would not be directly helpful in POS disambiguation, cf. Wilks and Stevenson (1998).

[4]For English, 45 POS categories exist in the Penn Treebank (Marcus et al., 1993) and 87 in the Brown corpus (Francis and Kucera, 1979). The universal POS tagging scheme defines only 12 coarse POS tags (Petrov et al., 2012).

But	the	issue	is	stickier	than	it	seems	.
CC	DT	NN	VBZ	JJR	IN	PRP	VBZ	.

Figure 4.3: An English sentence with parts-of-speech, taken from the WSJ part of the Penn Treebank.

nouns. From the point of view of automatic analysis, the POS tag assignment is often ambiguous. In the above example, for instance, "issue" is ambiguous between NN, VB (base verb form) and VBP (non-3rd person singular), and "but" is ambiguous between CC, IN and RB (adverb).

4.2.3 Syntactic parsing

Syntactic parsing is the task of automatically analyzing the syntactic structure of an input sentence. It is common to distinguish between parsing systems that output constituent (or phrase) structures and those that output dependency structures. The type of information captured is different in each case (Kübler et al., 2009): In constituent parsing—perhaps the most widely used and studied in computational linguistics—the task consists of grouping words into constituents (phrases) that are labeled by structural categories, like noun phrase (NP) and prepositional phrase (PP); in dependency parsing, on the other hand, we are interested in the predicate–argument structure, identifying head–dependent (or, parent–child) relations between words labeled by the function of the dependent toward its head, e.g. subject (SBJ) and object (OBJ). This difference is illustrated in Figure 4.4 and Figure 4.5.

A parsing system can be described with a schema depicted in Figure 4.6. It represents parsing as a task consisting of two stages: *(strict) parsing*, in which the goal is to generate several possible syntactic analyses for a given input, and *disambiguation* (or *parse selection*), which selects one among the ambiguous parses, usually by means of statistical modeling (Plank, 2011). When the parsing component is based on an explicit, formal grammar that is usually manually defined and used for generating a set of possible parses,

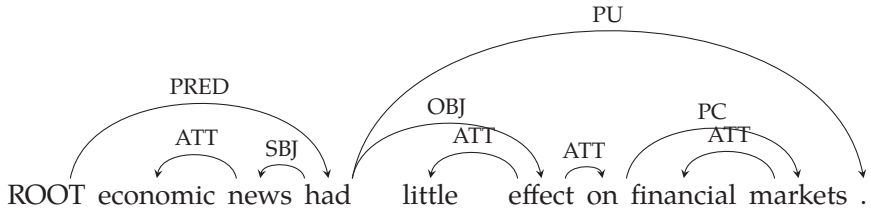


Figure 4.4: Dependency structure for an English sentence, reproduced from Kübler et al. (2009).

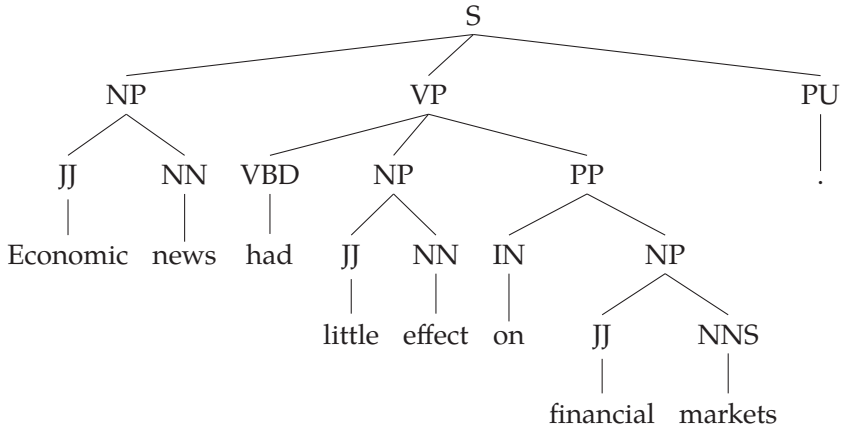


Figure 4.5: Constituent structure for an English sentence, reproduced from Kübler et al. (2009).

we can name the systems of this type the *grammar-driven systems*. Here, the role of the statistical disambiguation component is to use the learned model parameters for selecting a single, preferred syntactic structure. The Alpino parser for Dutch is an example of such a system. On the other hand, systems which lack an explicit grammar—or whose grammar is automatically induced—can be called *data-driven systems*: They are characterized by their reliance on corpora (annotated or not) to learn the grammar and the model. Representatives of this category are statistical dependency parsers,

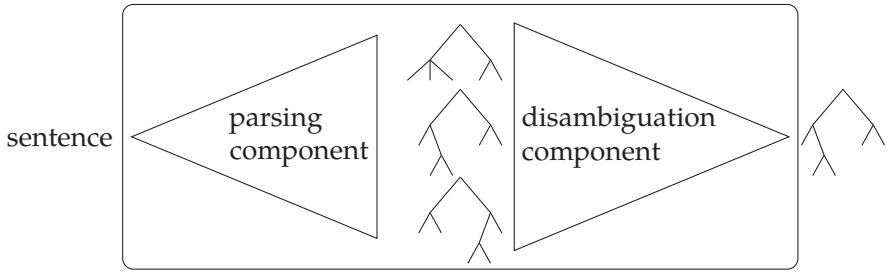


Figure 4.6: A two-stage view of a parsing system (reproduced from Plank (2011), drawing on lecture notes of Khalil Sima'an, University of Amsterdam).

such as Mate (Bohnet, 2010), Malt (Nivre, 2006) and MST (McDonald et al., 2005). These parsers allow easy integration of new learning features, including word representations. We introduce Alpino and the MST parser, as well as the gold standard datasets to train and evaluate them, in Chapter 5 and in Chapter 6. Note that in Chapter 5 we use Alpino to study the effect of concept-based word representations in reducing the lexical sparseness in parsing. In Chapter 6 and Chapter 7, though, our use of the two parsers is restricted to preparing the input texts for those representation-learning methods that rely on dependency structures.

4.2.4 Semantic frame identification

Frame-semantic parsing is the task of analyzing the textual predicate-argument structure such as the one shown in Figure 4.7, according to the theory of semantic frames (Fillmore, 1982). Two frames occur in this example sentence, HINDERING and CAUSE_TO_MAKE_PROGRESS. The first is evoked with a verb, and the second with a noun, although generally a single frame can be evoked with different lexical units belonging to different parts of speech. In English FrameNet (Baker et al., 1998), which is a resource containing the definitions and annotations of frames, specifies that HINDERING can be evoked by the lexical units "block (verb)", "delay (verb)", "inter-

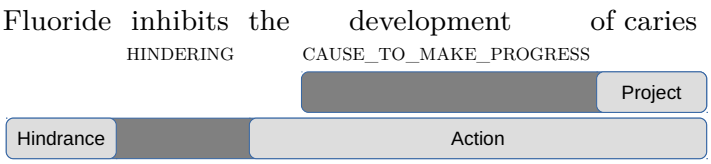


Figure 4.7: A parse with HINDERING and CAUSE_TO_MAKE_PROGRESS frames with respective arguments.

ference (noun)" and "repressive (adjective)", among others.[5] Similarly to WordNet, the frames stand in hierarchical relations, such as *inheritance* (a frame is a subtype of another frame) and *using*. The latter relation links to a frame of the presupposed parent frame that serves as background. In the above example, HINDERING presupposes the EVENT frame, and HINDERING is itself presupposed by DIFFICULTY. The HINDERING frame governs two syntactic dependencies, the nominal subject to the left and the nominal phrase to the right. These constitute the necessary participants in the frame, also called *core frame elements* (or arguments). The subject, "fluoride", represents a "Hindrance", and the object represents an "Action". The definition of the HINDRANCE frame includes an additional, third core element, "protagonist", but since the action in the realization of the frame shown above is a natural act, the protagonist is not instantiated. Other, non-core (also known as peripheral) elements can also occur in a frame, but none occur in the above example. These elements introduce non-essential information such as time, place and manner. For HINDERING particularly, several such non-core element types exist and could be realized in some other text, e.g. degree, duration and place.

As we have seen in the above example, one elementary distinction in the FrameNet scheme is between the annotation of frames and elements

[5] Any of these lemma-POS pairings may evoke also other frames. In principle, each sense of a lemma-POS pair describes a new frame (Ruppenhofer et al., 2006).

participating in it. This duality is also encountered in the context of automatic frame semantic analysis (Das et al., 2014). In NLP, the task including the analysis for both components is usually referred to as *frame-semantic parsing*. This involves the following: *a*) the identification of targets, i.e. predicating words, typically realized by a heuristic component; *b*) the classification of semantic *frames* of predicates in a sentence[6]; and *c*) the identification of frame *arguments* participating in these events, which can also be seen as a semantic role labeling task. In this thesis, we adopt the second subtask of semantic frame identification. The goal is to measure the capability of word representations to better disambiguate between the frames by differently representing different word senses. Compared to NER, in which the classification decisions apply to a relatively small set of words, the problem of semantic frame identification concerns making predictions for a broader set of words, including verbs, nouns, adjectives and sometimes even prepositions. The semantic frame identification can thus give us more complete information about the quality of the word representations.

[6]Since a polysemous predicate can participate in different frames, this subtask is similar to performing word-sense disambiguation.

PART II

Reducing lexical sparseness with semantic classes

CHAPTER 5

Using wordnet concept classes in a grammar-driven parser

Bilexical information as obtained from selectional preferences is known to be helpful in syntactic parsing. However, using the lexical information alone in form of word types or lemmas is bound to lead to lexical sparseness. A possible solution is to adopt more general representations of words acquired with word concept classes. In this chapter, we study the problem of lexical sparseness in a syntactic parser for Dutch by first analyzing the parser's performance on each feature type described by a dependency relation label. We then shortlist those feature types that would benefit from class-based generalization the most. Finally, we report the results of a generalization using word classes obtained from a wordnet inventory with varying degrees of generality. Although our method corrects several previously misparsed cases, it does not improve the accuracy overall.

5.1 Motivation

Several syntactic parsers use lexicalized information in either strict parsing or parse disambiguation steps (Bikel, 2002; Collins, 2003; Charniak, 2000; van Noord, 2006). A bilexical feature, such as the one to learn that a particular verb tends to occur with a specific object, e.g. "drink a beer", is modeled from large corpora, yet for many instances during parsing, such bilexical information is missing. For example, consider that the word "beer" has been substituted with a less frequent drink such as "daiquiri". It is less likely that in this case the model will hold reliable information for "drink a daiquiri". To address this issue of lexical sparseness, we can apply a generalization procedure that will work by grouping multiple lexical units that are similar in meaning. Such semantic generalization has been shown to be effective in the works of Agirre et al. (2008), Henestroza Anguiano and Candito (2012) and MacKinlay et al. (2012), all of which acquire sets of word senses or broader semantic categories from human-built lexical inventories such as the English WordNet (Miller, 1995). The question we ask ourselves in this chapter is

- whether semantic classes applied to the existing bilexical model of a syntactic parser of Dutch can additionally reduce lexical sparseness, and as a result, lead to an additional improvement of parsing accuracy.

Specifically, we aim to:

- quantify the relationship between successful parses and the availability of bilexical information in the existing disambiguation component of the parser,

- identify empirically those feature types described by dependency relations labels for which the semantic generalization of participating words is most needed in parsing,

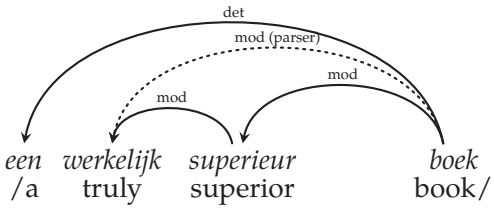


Figure 5.1: A sentence excerpt with human-annotated dependency structure. The dotted line represents an incorrect attachment by the parser.

- provide the results of a generalization procedure for the disambiguation component using semantic classes extracted from a human-built inventory.

To further illustrate the need for generalization in syntactic parsing,[1] we provide an example excerpt from a Dutch sentence with its corresponding dependency analysis in Figure 5.1. The parser wrongly attaches the adverb "werkelijk" to the noun "boek", whereas the correct analysis would connect "werkelijk" to the adjective "superieur". Here, the reason for the wrong attachment can be attributed to the missing bilexical information in the disambiguation model of the parser. Naturally, it is impossible to expect to find innumerable bilexical cases like ⟨"werkelijk", "superieur"⟩ in the disambiguation component. Intuitively, it should be possible to resolve this case of ambiguity between modification of adjective and modification of noun, as well as other types of ambiguity, if we abstract away from the concrete word level. For example, in the Cornetto lexical semantic inventory for Dutch, the word "superieur" shares the semantic class (synset) with the word "goed" ("good"). Since the model already includes the information about ⟨"werkelijk", "goed"⟩, the semantic class generalization in this case means a successful model lookup and can possibly result in a correct parse.

[1] A more general discussion of lexical sparseness can be found in section 2.1.

Previous studies on this topic have typically focused on generalizing words regardless of the dependency relation in which they appear (see section 5.4 for a detailed account of related work). Our approach differs by first identifying those feature types described by dependency relation labels for which the parsing accuracy is reduced. The premise of our approach is that generalization should not be imposed where the lexical sparseness is not severe. Following a statistical analysis, we list the feature types that are most likely to be helpful once the model has been extended with semantic information (section 5.3). We experiment with three types of semantic representations (section 5.2.5), which vary in their degree of generality. Whenever the bilexical preference is lacking in the standard model, our enhancement model would simply look up a generalized version of the lexical pair. We show that this leads to parsing improvement in certain cases, but it is noisy in general and causes also several deteriorated parses.

5.2 Experimental setup

In this section, we first present the parser and the training and testing resources. We also present the resource from which we retrieve semantic classes. We then introduce the method for analyzing the bilexical model usage and discuss the disambiguation method, the levels of semantic representation and how the new information is introduced in the modeling part of the parsing process.

Alpino. We can describe Alpino (Bouma et al., 2001; van Noord, 2006) concisely as a parser featuring a head-driven phrase structure grammar (HPSG) and a maximum entropy disambiguation component. The grammar is augmented to output dependency structure in LASSY format[2]. This format includes both functional annotation (dependency label) and syntactic category information, as well as the part-of-speech (POS) categories. Dependency relations hold between sibling nodes; one of the nodes is assigned the "hd" label, for head of the relation. According to the LASSY an-

[2] http://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf

notation scheme, the syntactic structures are not always proper trees, since more than one parent per word is allowed. The output of Alpino's parsing component for a sentence is a packed parse forest. The best parse is obtained through maximum entropy modeling with a best-first beam-search algorithm. The details thereof can be found in van Noord (2010).

We evaluate the parser's performance with the concept accuracy (CA) measure, which is a mean of per-sentence minimum of recall and precision. In practice, the measure gives similar results to labeled attachment score (LAS) (van Noord, 2009b), which is the percentage of correct dependencies, counting a dependency instance as correct if both the attachment and the label are correct. The main reason for using the CA measure is to relax the single-head constraint of the LAS measure. Alpino achieves an accuracy of around 90% on general newspaper texts (van Noord, 2010).

Bilexical disambiguation component. The disambiguation component of Alpino includes a large number of features, divided in several feature types. Only some of these feature types measure selectional information about word pairs (*the bilexical component*), but we list all types for completeness in Table 5.1.

Description	Types
Grammar rule applications	r1, r2
POS tags	f1, f2
Dependency features	dep23, dep34, dep35, depprep, dep-prep2, sdep
Ordering in the middle field	mf
Unknown heuristics	h1
Bilexical information	*z_dep35*, *z_hdpp*, *z_appos_person*
Other	appos_person, s1, p1, in_year, dist

Table 5.1: Alpino feature types.

The feature types under *bilexical information* capture the degree of association between a pair of words in a dependency relation. For our purpose, it will be useful to further divide the bilexical feature types along the de-

pendency relation label and the POS tag of the head word that can appear in the relation. In this way, we will be able to distinguish in our analysis between cases of modification of nouns ("z_dep35(noun, hd/mod)") and modification of adjectives ("z_dep35(adj, hd/mod)"), to name just one example. Reusing the phrase from Figure 5.1, we can fully instantiate the feature types in the following way:

(5.1) ⟨superieur, adj, hd/mod, noun, boek⟩

(5.2) ⟨werkelijk, adv, hd/mod, adj, superieur⟩,

where the first element is the dependent word, which is followed by its POS tag; the middle element is the dependency relation; the before-final and final elements are the POS tag and the head word, respectively. As we have just seen, the dependency relation label and the POS tag of the head can constitute the broader specification of a feature—we will refer to such feature types as *partially instantiated feature types*. So, we will treat "z_dep35(adj, hd/mod)", for example, as one such type. In addition to the z_dep35 feature type, we have two other bilexical feature types: z_hdpp and z_appos_person. The z_hdpp type captures second-order effects by considering the node of a prepositional phrase, as in the partially instantiated type "z_hdpp(noun, hd/pc)", which stands for a noun head in combination with a head word of the prepositional complement. For illustration, consider

(5.3) organisaties die zich met de inburgering van immigranten bezighouden
"organizations dealing with naturalization of immigrants",

from which we can extract the ⟨inburgering, noun, hd/pc, van, houd_bezig, verb⟩ feature instance, where the first element is the head word, which is followed by its POS tag and the dependency relation label; the last three elements are the preposition, the node of the prepositional complement and

its POS tag.[3] The third type, z_appos_person, pertains to named-entity related features, but since it rarely occurs in the disambiguation model, and since the semantic classes including named entities are uncommon in our lexical semantic inventory for Dutch, we discard it from further consideration and work only with z_dep35 and z_hdpp feature types.

Every feature instance in the bilexical part of the disambiguation model is associated with a normalized pointwise mutual information score with a frequency threshold for inclusion of > 50. The frequency threshold is applied to avoid the unreliable mutual information scores for infrequent events, as well as to keep the size of extracted instances manageable. The best parse is selected by a linear combination of features with their learned weights (van Noord, 2007):

$$\hat{y} = \arg\max_{y} \sum_{i} w_i f_i(x, y), \tag{5.1}$$

where the mutual information score is one feature function (f_i) among many for the the sentence–parse pair (x, y). The total number of partially instantiated bilexical feature types is 35. The number of all feature instances in the disambiguation component in our experiments is around 3.3 million. It is worth noting that the association information is obtained from parsed structures, which means that the training corpus is first parsed (using the existing disambiguation component), and then the new association scores are acquired. The method thus represents a self-learning approach to obtaining lexical information. van Noord (2010) shows that the incorporation of the bilexical information into parsing improves over the parser without it, with the advantage ranging between 0.4–0.5 CA score. In that work, a corpus of Dutch texts amounting to around 500 million words is used to learn the bilexical preferences.

[3] Note that the z_hdpp instance captures different information from the z_dep35 examples above.

5.2.1 Corpora

The statistical analysis that will serve us as the basis for the identification of promising partially instantiated feature types is carried out on Lassy Small, a hand-annotated corpus of Dutch. The corpus contains around 1 million words corresponding to 1.3 million dependency instances for a large variety of texts like newspapers, Wikipedia, websites, fiction etc. (van Noord, 2009a).[4] An automatically syntactically annotated part of the preliminary version of Lassy Large[5] is used for training the lexical association model of Alpino. This corpus contains about 500 million words. We test our method on the Alpino Treebank (van Noord, 2006)[6], which amounts to ≈7,000 sentences, and is a collection of newspaper texts from the Eindhoven corpus, and parts of Lassy Small (≈4,000 sentences).

5.2.2 Statistical analysis of parser's performance and disambiguation model lookup

We are interested in studying the relationship between the presence of bilexical information in the disambiguation model and the correctness of the parses. We would like to find candidates that are promising for generalization, where by candidate we mean any partially instantiated feature type. A candidate should thus be the one for which the bilexical information in the disambiguation model is often missing, which then tends to negatively affect the parsing outcome.

We carry out the statistical analysis both on the level of *sentence* and on the level of *individual instances* (dependencies). In the first scenario, the parsing accuracy can be measured as per-sentence concept accuracy (CA), while the model presence can be measured as the proportion of instances for that sentence found in the bilexical model. In the second case, both variables need to be categorical (binary), as an instance can either be in

[4] http://tst-centrale.org/producten/corpora/lassy-klein-corpus/6-66

[5] http://tst-centrale.org/producten/corpora/lassy-groot-corpus/6-67

[6] http://www.let.rug.nl/~vannoord/trees/

the model or not, and the parse can be either correct or not. We therefore use several statistical tests to take these differences into account. For the sentence-level analysis, we use the Spearman non-parametric correlation test, while for the instance-level analysis we use the Cramér's phi (ϕ_c) association measure (interpretable just as a correlation score, i.e. ranging between 0—no association—and 1—perfect association). In addition, we will report the odds-ratio score (Gries, 2009) due to its ability to indicate how the likelihood of one variable level changes in response to a change of another variable's level. For example, we will be able to say how much more likely a correct parse is when the bilexical preference is found in the disambiguation model (call it E for a positive event). The odds ratio is expressed simply in terms of the probabilities for the positive event (p_E) and the negative event ($1 - p_E$):

$$odds = \frac{p_E}{1 - p_E}. \tag{5.2}$$

Note that all measures can be applied to all candidates grouped, which gives an indication of effectiveness of the bilexical disambiguation model in general, but they can also be obtained on each candidate separately.

5.2.3 Lexical semantic inventory

Cornetto[7] is a lexical semantic inventory for Dutch resulting from a merge between Referentie Bestand Nederlands (RBN, a collection of lexical units) and Dutch WordNet (Vossen et al., 2013). It includes more than 92,000 form-POS pairs, described in terms of lexical units, synsets and other criteria. In wordnets, words are typically specified by a coarse POS label (hence the term form-POS pair), e.g. "file-n" for "file" as a noun and "file-v" for "file" as a verb. The synsets represent sets of synonymous form-POS pairings. When looking at a particular meaning of a form-POS pair (i.e. a combination of the form, the POS label and the synset identifier), we talk about

[7] Part of it is presently available as Open Source Dutch Wordnet (Postma et al., 2016).

a lexical unit. Table 5.2 lists some statistics of Cornetto, version 2.

Type	All POS	Nouns	Verbs	Adjectives	Adverbs
Synsets	70,370	52,845	9,017	7,689	220
Lexical units	119,108	85,449	17,314	15,712	475
Form-POS pairs	92,686	70,315	9,051	12,288	1,032

Table 5.2: Cornetto statistics per part-of-speech category.

Cornetto also includes coarse semantic categories, called *semantic types*, which provide a very general description analogous to English WordNet semantic files, also known as super-senses (MacKinlay et al., 2012; Agirre et al., 2008). Around 20 semantic types are found in Cornetto. We list the ten most frequent ones in Table 5.3.

Semantic type	POS
nondynamic	noun
action	verb
artefact	noun
human	noun
dynamic	noun
abstract	noun, adjective
place	noun, adjective
concrete	noun
emotional/mental	adjective
process	verb

Table 5.3: 10 most common semantic types in Cornetto.

The types are POS-dependent, meaning that a verb, for example, can be described with the "action" class, but not "place", which is reserved for nouns and adjectives. The semantic type information is attributed in Cornetto to about one half of all lexical units. We refer the reader to Cornetto user documentation[8] for more information on semantic types.

[8] `http://cornetto.clarin.inl.nl/help/D9-Cornetto-Documentation.pdf`

For the purpose of determining the generality of a synset, we treat Cornetto as a set of digraphs (one for each part of speech), with nodes constituting synsets and arcs constituting hypernymic relations. Since the graphs have a tree-like structure, we use the term *leaves* to denote the most concrete synsets with no incoming arcs (hyponyms), and *top* to denote the most abstract node with no outgoing arcs. We use the Information Content (IC) measure as defined in (Sánchez et al., 2011):

$$IC(s) = -log\frac{(|\mathcal{L}_s|/|\mathcal{S}_s|) + 1}{|\mathcal{L}| + 1}, \tag{5.3}$$

where $\mathcal{L}$ are the leaves of the hierarchy, $\mathcal{L}_s$ are the leaves reachable from a synset s, and $\mathcal{S}_s$ are the subsumers of s (more general synsets located higher in the hierarchy). Intuitively, when s is located high in the hierarchy, many leaves can be reached from it, hence $\mathcal{L}_s$ will be large, and the final IC score will be low. Conversely, more specialized synsets, located lower in the hierarchy, bear more information content.

5.2.4 Disambiguation method

To circumvent the problem of sense ambiguity when mapping classes to word forms, we use the first-sense heuristic. Although this technique yields wrong classes in certain number of cases, it has been shown to perform well in the works of Agirre et al. (2008, 2011) and McCarthy et al. (2004). Unlike the sense ordering in English WordNet (McCarthy et al., 2004), the senses in Cornetto are not ranked by frequency but by salience. The reason for this can be tracked to the construction criteria of RBN on which Cornetto is based. In RBN, various lexicographic criteria were considered when ordering the senses, frequency in corpora being only one of them. We refer the reader to Martin et al. (2005) for a more detailed account of these criteria.

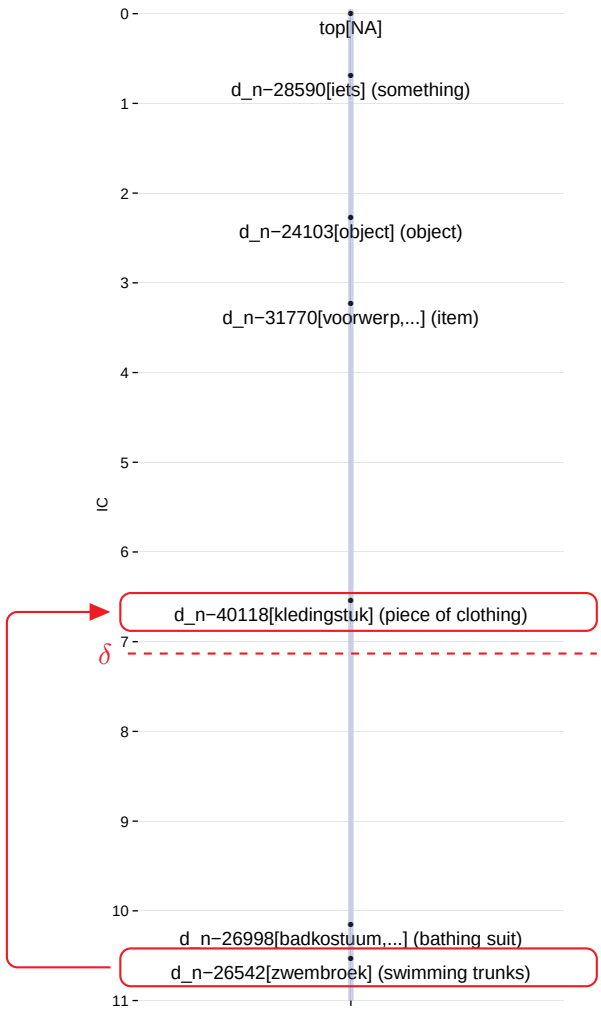


Figure 5.2: Obtaining an intermediately-grained representation for the word “zwembroek” using an IC threshold value δ. English translation in parentheses.

5.2.5 Levels of semantic representation

We explore semantic classes of varying degrees of generality. The most straightforward way to represent a word-POS pair using Cornetto is to use its immediate, first-listed synset. We will treat the immediate synset as a *fine-grained* semantic representation of a lexical unit, as it is the most concrete representation type used in our work. Synsets provide a generalization whenever they group more than one lexical unit. However, a singleton synset in fact adds specificity to the form-POS instance, by licensing only a single sense of the instance. This duality has been also noted in the work of Bikel (2000) on word-sense disambiguation and parsing.

In obtaining *intermediately-grained* representations, we make use of the previously introduced Information Content. We first compute the IC score for a synset s. Then, if $IC(s)$ exceeds a threshold δ, s is too specific, therefore we would like to consider a more general synset. The suitable generalized synset s_{gen} is the one closest to, but smaller than δ. This is exemplified in Figure 5.2. However, when it is already the case that $IC(s) < \delta$, the synset is located sufficiently high in the hierarchy, therefore no generalization is needed. We set the value of δ manually, by inspecting hypernymic paths of various lexical units. At this stage, no empirical optimization of δ is attempted. For the experiments described below we set δ to 6.

The most general representation type we consider in this work is the semantic types, already described in section 5.2.3.

5.2.6 Application of semantic classes to disambiguation model

The incorporation of semantic classes is performed on the bilexical part of the disambiguation component without retraining. The semantic class identifiers are assigned to words in all the feature instances belonging to one of the selected partially instantiated feature types. Then, the MI scores are calculated. In order to improve the mapping between word roots in the bilexical part of the model and word lemmas in Cornetto, we preprocess the lemmas in Cornetto with the Alpino lexical analyzer.

The new information is made available as the data that is auxiliary to the bilexical part of the disambiguation model. In this way, it is possible to use the generalized model in a back-off manner. The instances extended with semantic classes are only considered after a failed lookup in the bilexical model.[9] We use the back-off strategy because we see generalization as potentially helpful only when lexical sparseness is too severe. When this is not the case, using bilexical preferences directly is known to be effective in its own right (van Noord, 2010).

5.3 Empirical study

5.3.1 Relationship between parsing accuracy and disambiguation model lookup

We carry out this analysis on a test dataset of 62,944 sentences. Correlating the two variables—the *CA* score and the *in-model proportion*—reveals a mild positive relationship of Spearman $\rho_s = 0.239$. The correlation is visualized in Figure 5.3. Each light-gray dot represents a value for one sentence. When multiple sentences achieve the same value, the overlapping dots yield a darker-colored effect. We can observe in the plot that the majority of points are located in the upper part, with CA values above 50. Also, the points tend towards the right half of the plot, which is a sign that on average more than a half of the dependency instances would be found in the disambiguation model.[10] The blue curve is a *loess* (for "locally weighted scatter-plot smoothing") curve which summarizes the scatter: Low CA values correspond to a lower proportion of in-model information, but high CA values correspond to a higher proportion of in-model information.

[9] We have also experimented with using semantic classes by default and backing off to bilexical model, but this led to a reduced performance.

[10] The plot also reveals an artifact of the experimental setup, displayed as unevenly distributed dots. The distribution is disrupted at regular intervals, yielding an effect of parallel dotted lines. This is explained by the fact that both measures use proportions (for example, a sentence of length 3 can only yield 4 possible values for the in-model variable), which are not continuous.

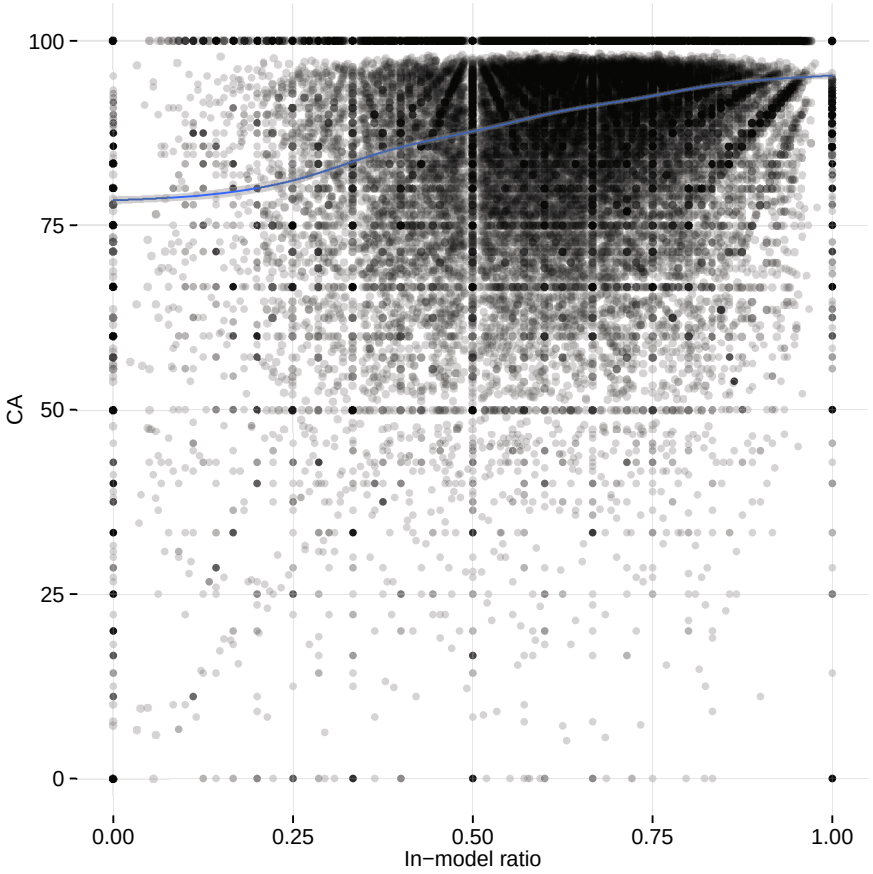


Figure 5.3: A plot of the in-model ratio against the CA scores from the sentence perspective. The loess curve summarizes the variation in the data.

We now correlate the in-model proportion and parsing accuracy per instance. So, for a single instance, both the in-model proportion and parsing accuracy are treated as binary variables. The Pearson's chi-square test confirms ($p < 0.001$) that parsing accuracy on instances which *were* in the model differs to instances *not* in the model. The odds ratio on all feature

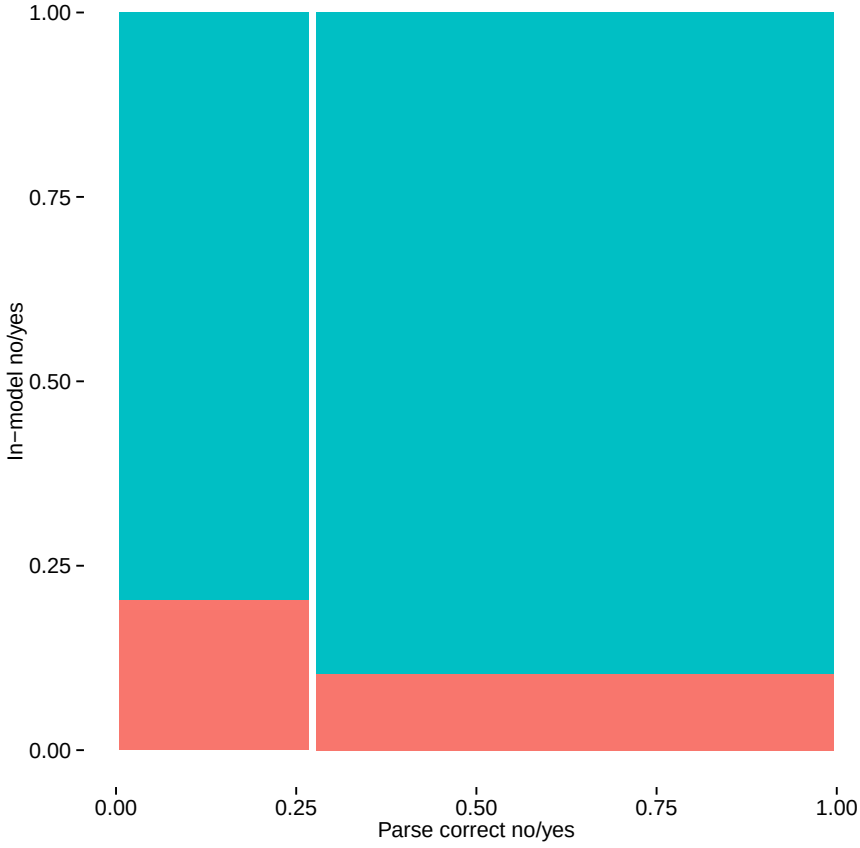


Figure 5.4: Mosaic plot of in-model proportion and parser accuracy.

types is 3.6, which means that a parse is 3.6 times more likely to be incorrect when the bilexical feature is missing. Figure 5.4 pairs our variables of interest. We can see that incorrect parses are less frequent than correct ones, but they are characterized by a higher number of unsuccessful model lookups.

Type	Found in corpus	Found in model	%
z_dep35(verb, hd/vc)	41798	40877	97.8
z_dep35(verb, hd/mod)	96609	87591	90.7
z_dep35(prep, hd/obj1)	135645	115582	85.2
z_dep35(verb, hd/su)	113658	86640	76.2
z_dep35(adj, hd/mod)	11828	8859	74.9
z_dep35(noun, hd/mod)	133925	86430	64.5
z_dep35(noun, cnj/cnj)	18848	5512	29.2
z_hdpp(adj, hd/mod)	3254	569	17.5
all	924783	672535	72.7

Table 5.4: In-model proportions for a selection of partially instantiated feature types.

The partially instantiated feature types which we would like to enhance with semantic classes are selected according to the following criteria: the odds ratio, the Cramér's phi association coefficient (both measuring effect size) and the number of out-of-model instances that were parsed incorrectly. Table 5.4 shows the in-model proportions across the commonest types. The in-model proportion is the highest for cases of verbal complements (hd/vc), in which the complement can only be introduced with a limited set of words. For our purpose, the prevailing type of error observed in incorrect parses should be wrong attachment. We therefore discard those types which mostly include other error types, such as incorrect dependency relation labels resulting from limitations in the grammar or from non-standard language phenomena. The final selection consists of types listed in Table 5.5 and described briefly here:

- **modification of the adjective** occurs mostly with another adjective

or an adverb. The challenging aspect for the parser is to choose the correct attachment site.

- **nominal coordination**—at least one of the coordinated words must be a noun. Here too, the prevalent type of mistake is an incorrectly attached coordinated element. Less frequently, the construction is not recognized as such by the parser. This happens especially when the coordination is less typical, as with the conjunction "evenals" ("just as").

- **modification of the noun**—nouns are most commonly modified by the prepositions introducing prepositional phrases. Adjectives and other nouns can act as modifiers as well. Just as with the previous types, incorrect attachment site is the main source of errors. A manual analysis reveals that parsing mistakes of this type occur often in long sentences with heavy modification.

Type	Instances	Odds	ϕ_c	Out-of-model #/%
z_dep35(adj, hd/mod)	11828	2.653	0.20	1213/10.3
z_dep35(noun, cnj/cnj)	18853	2.042	0.12	3192/16.9
z_dep35(noun, hd/mod)	133925	1.962	0.11	8003/6.0

Table 5.5: A shortlist of partially instantiated feature types with respective statistics.

Note that all the types have relatively low association coefficients because the bilexical information is only a supplementary source of features in the parse selection component.

5.3.2 Generalization with semantic classes

For the selected types from Table 5.5, any noun, adjective or adverb participating in them will undergo generalization with fine- and intermediate-grained classes. The semantic types from Cornetto, however, can only be

obtained for nouns and adjectives since they are not defined for adverbs. We show in Table 5.6 how the inclusion of semantic classes affects the overall parsing accuracy. The configuration using immediate synsets (SYN) does not result in an overall improvement of parsing accuracy. The average performance with ten-fold cross validation levels the baseline parser configuration at 90.46 % CA. The SYN configuration leads to 33 improved dependency instances, however it also introduces 29 incorrect dependency instances. There are several reasons why the number of actual parse modifications here is small. First, the access to the new information is attempted relatively rarely, only after a failed model lookup. Second, Cornetto synsets could only be mapped to around 60% of the form-POS pairs encountered during parsing. Third, the success of finding a generalized instance having at least one successfully mapped synset ID is 7.85%. This low number means that synsets cannot generalize sufficiently, which is understandable—many synsets include only one lexical unit.

Method	% Found	Improved	Deteriorated	CA new	CA old
SYN	7.8	33	29	90.46	90.46
ST	62.1	178	299	90.35	90.46

Table 5.6: Parsing results obtained by generalization with semantic classes. SYN: immediate synset, ST: semantic type, Found: instances found in back-off, Improved: incorrect-to-correct parse modification, Deteriorated: correct-to-incorrect parse modification. Total number of test sentences: 11,053.

In the experiment with coarse semantic types (ST), the results are even worse: The parsing accuracy drops to 90.35%. Although the number of correctly introduced parses increases in comparison with SYN (to 178), the number of modifications which result in an inaccurate parse increases as well (299). The new information thus clearly overgeneralizes. It is possible that the generalization we obtain with only 20 semantic types for Dutch, as opposed to 45 types in English WordNet (Agirre et al., 2008), influences the resulting precision. The results in Table 5.6 show how a better recall (%

found) from the enhanced model corresponds to a deteriorated precision (# *deteriorated*).

The best performing type among the 3 identified types was nominal conjunction. We applied the method of intermediate granularity (INT) to this type only. The results for nominal conjunction are shown in Table 5.7. INT method does introduce a higher number of modifications than SYN, but again at the expense of precision. Further experiments would be needed in order to confirm whether INT is a better performer than the ST method. A manual inspection of instances that were parsed correctly as a result of the incorporated semantic classes confirms our reasoning on the motivating example from Figure 5.1 for all three dependency types. Consider the following sentence encountered during testing:

(5.4) De afgelopen week werden er in de *Utrechtse Camera bioscoop* elke dag Tarzanfilms gedraaid.
Last week, Tarzan films were shown daily in the Camera cinema in Utrecht.

Method	Improved	Deteriorated	CA new	CA old
SYN	7	2	90.47	90.46
ST	20	26	90.45	90.46
INT	16	19	90.45	90.46

Table 5.7: Parsing results for the "z_dep35(noun, cnj/cnj)" type. *INT*: class of intermediate granularity (based on IC scores with $\delta = 6$).

The attachment of the word "Utrechtse" is ambiguous between "Camera" and "bioscoop" ("cinema"). The pair representing the correct attachment, ⟨"Utrechtse", "bioscoop"⟩, was not found in the bilexical model and was not attached correctly by the parser. The word "Utrechtse" was successfully substituted with a synset representing place, which then enabled a successful parse. In contrast, incorrect parses are mostly introduced either because the head–dependent pair for which the model provides support

stands for a wrong attachment, or because the relation label in the treebank is different (e.g. apposition instead of modification).

5.4 Additional related work

Research on parsing improvement with semantic generalization can be roughly divided into approaches that introduce lexical semantic information from human-built resources and the approaches which acquire semantic classes distributionally, e.g. by clustering. Here, we introduce the existing work of the first type, since it is the most relevant to our work.

One of the earliest attempts to bring lexical semantic information in parsing is the work of Bikel (2000). He incorporates WordNet classes into a lexicalized generative PCFG model for English, but without major improvements. Bikel also considers several levels of generality but does not try to determine the optimal level.

Xiong et al. (2005) use resources similar to WordNet for Chinese in order to retrieve sense information; they use three levels of generalization: the immediate synset as well as two distinct hypernym synsets. The new class-based information is incorporated as selectional preferences in a generative lexicalized model, which improves over their baseline model. Fujita et al. (2007) develop a parse selection model for Japanese, which in its best configuration uses both syntactic and semantic features. The semantic features are based on dependencies extracted from semantic representations of sentences. Just like in Xiong et al. (2005), the dependency triple elements are substituted by senses and hypernym classes at various levels. Based on semantic dependencies and valency features, they achieve a substantial improvement over their best syntactic model.

Agirre et al. (2008) experiment with two lexicalized parsing models (Charniak, 2000; Bikel, 2004) by mapping semantic classes to the training data. They evaluate the extended models on both general parsing and PP-attachment disambiguation. Three levels of semantic representation are incorporated: synsets, coarse semantic files and hybrid word–semantic file

representations. The semantic files and the first-sense heuristic for disambiguation turn out to be good performers in most of the experiments. The maximum performance gain observed is 1.1% (in F-score) in general parsing and 5.6% in PP-attachment. A more recent work by Agirre et al. (2011) reproduces these findings, but this time including semantic classes as features into a data-driven dependency parser (Nivre, 2006).

Henestroza Anguiano and Candito (2012) introduce probabilistic generalization features for dependency parsing of French. In their work, a word is represented as a probability distribution over a space of generalized classes, which can be either lemmas, clusters or synsets. They observe slight improvements on the French Treebank and an out-of-domain medical corpus using the first-sense synsets from French EuroWordNet. Since they obtain somewhat higher scores when testing on the out-of-domain data, they argue that parsing improvement is more likely to be successful when there is a lexical divide between the training set and the testing set. Similarly, MacKinlay et al. (2012) show mixed results in HPSG parse selection using the English Resource Grammar. The authors observe no improvement in including synset features from hypernym paths. The best performing representations are found to be semantic files, which lead to a small reduction of error rate.

Clark (2001) learns selectional preferences based on classes obtained from WordNet and evaluates them in parse selection. Although he fails to improve the base selection component of the Carroll and Briscoe (1996) parser, the work is relevant to us since it also addresses the issue of selecting a suitable level of generalization. In contrast to other works we have just referred to, Clark (2001) investigates a more principled way of controlling the generalization level. His procedure involves a chi-square test with the significance level acting as a parameter for controlling the extent of generalization.

The work of Kiperwasser and Goldberg (2015) is close to ours in that the bilexical statistics are based on automatically-parsed data. The extracted statistics are included as features in a graph-based dependency parser.

Compared to the related work, our approach differs in that we build a generalization component on top of a bilexical disambiguation component that has already been included successfully in the syntactic parser. Additionally, in our work we do not apply semantic classes indiscriminately to words in all dependency types, but enhance only specific dependency types that are identified empirically. We observe that bilexical information is differently useful for different dependency types, which are in turn differently problematic for the parser. This constraint can in principle avoid applying the new information to dependency types for which the bilexical model performs well. Our work further differs by including semantic information not into full parsing, but in the existing parse selection component. Additionally, we enhance only the bilexical part of the disambiguation component, which limits the room for improvement.

5.5 Conclusion

We have presented an investigation of the lexical sparseness in the parsing of Dutch, and a technique relying on semantic classes to reduce it. The enhancement of the existing bilexical model for parse selection with semantic classes does not result in improved parsing accuracy, which somewhat mirrors the results of MacKinlay et al. (2012) and Bikel (2000). We would like to emphasize that in our case the bilexical information had been already explored and included in the Alpino parser, which makes the baseline against which we compare stronger. In the experiment in which immediate synsets are used with a back-off strategy after a failed lexical lookup, the number of correct parses is higher than the number of incorrect ones, but the total number of modifications is too low to be strongly reflected in the overall parsing accuracy.

We believe that the impact of our method is limited, firstly, because the parser commits errors of different types, with incorrect resolution of attachment ambiguities being just one. Often, an incorrect parse occurs because of a limitation in the lexicon or grammar, which does not foresee

a specific construction (which can then lead to an incorrect dependency relation label); because of the effect of long sentences or non-standard language phenomena (e.g. in the case of a misspelling); or because of incorrect or inconsistent gold annotation (cf. also Manning (2011)).

Because the synsets are fine-grained, they do not effectively reduce sparseness. Immediate synsets lead to more improvements than deteriorations because they represent a specialization (one sense of a word) so the learned selectional preferences are more accurate, too. In the light of previous research, it is surprising that the coarse semantic types performed worse than synsets. This could be at least partly attributed to a different specification and number of semantic types in English WordNet and Dutch Cornetto. The experiment in which we choose the suitable level of representation by measuring synset generality gives slightly better results than that of semantic types, although further work should be performed to assess the effect of the threshold parameter on the results. The fact that the number of parse modifications is low can be explained by Cornetto coverage and the back-off strategy. Our results suggest that higher levels of generalization yield a higher number of parse deteriorations. Another factor contributing to deteriorations could be the selection of incorrect senses for a given word form. It is possible that the first most prominent sense in Cornetto is often not the most frequent sense. Currently, the form of Dutch SemCor which would allow us to reorder Cornetto senses by frequency does not yet exist (Vossen et al., 2013). However, this is unlikely to matter for senses which are difficult to distinguish and for high levels of representation with higher chance of converging senses. Further, as noted at the end of section 5.4, the room for improvement in our experiments is smaller compared to studies enhancing full parsing or selection components more generally. It seems that the degree of lexicalization of the parser partly determines the impact of generalization techniques too. For example, Plank (2011) shows that removing lexical features from Alpino's selection component affects the performance relatively little compared to some data-driven parsers whose performance can drop as much as 10.5% when unlexicalized.

The drop percentage might be an indicator of the expected final impact of generalization.

In the next part, we develop distributional models for inducing semantic classes, which could effectively tackle the problem of resource coverage—note that around 40% of words could not be mapped to semantic classes because they were not found in the lexical semantic inventory. Distributional approaches in general can have further advantages, namely an increased adaptability for out-of-domain parsing, possibility to vary sense granularity and sense ranking, and the ability to disambiguate word senses. We will propose two models from which we can obtain lexical features which can be used for a variety of tasks.

PART III

The role of syntax in models of word representations

CHAPTER 6

Dependency Brown clustering

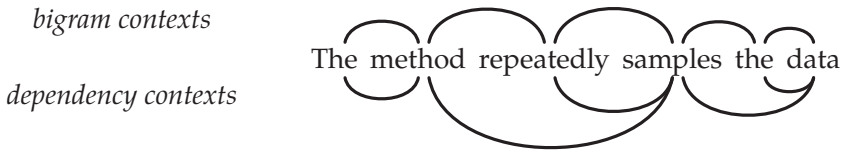
We present an effective modification of the popular Brown et al. (1992) word clustering algorithm using a dependency language model. By leveraging syntax-based context, the induced clusters outperform the standard Brown clusters in a wordnet-based similarity task for Dutch. We find the improvements to be stable across parameters such as number of clusters, minimum frequency and granularity. Another practical advantage is that dependency Brown clustering achieves a desired clustering quality with less data, which means shorter clustering times. At the end of this chapter, we also look at the clustering quality when the training data consists of the instances from specific dependency relations and find that certain relations provide better context, which leads to further improvements over the standard Brown clustering.

6.1 Motivating the use of syntactic context

Semi-supervised approaches have been successful in various areas of natural language processing. Among a plethora of clustering techniques,

Brown clustering (Brown et al., 1992) is popular for its conceptual simplicity, available implementations (Stolcke, 2002; Liang, 2005; Klusáček, 2006), and because the resulting word clusters have been shown to be helpful in several NLP tasks. The induced clusters represent syntactic and semantic *generalization* of words, which can be helpful in dealing with word sparseness.

The Brown clustering groups words distributionally, based on shared context. However, only immediately adjacent words are taken into account, which is a known limitation (Koo et al., 2008; Sagae and Gordon, 2009; Grave et al., 2013). To give an example, even though verbs constitute informative contexts for object nouns, they might not always be immediately adjacent to the object nouns. However, by extracting the contexts from dependency relations, we get immediate access to these verbs. The difference between the two types of contexts can be illustrated with the following example:



The bigram context thus fails to capture the relation between the object "data" and the predicate "samples", as well as the one between the subject "method" and the predicate. Furthermore, the dependency representation rightly ignores some of the less informative contexts coming from immediately adjacent words. For example, there is no relation between the predicate "samples" and the article "the" to the right.

Given the above, it might be preferable therefore to induce word clusters based on the dependency relations in which the words occur. In section 6.3, we present how this relates to the standard, bigram-based Brown clustering algorithm, and we modify the implementation by Liang (2005) to perform dependency clustering. We evaluate the induced clusters in

a wordnet-based similarity experiment. Dependency clustering yields superior clusters for Dutch across different settings of parameters such as number of clusters, frequency threshold and level of granularity. Learning-instance selection on the basis of specific dependency relation labels further improves the clustering quality. The proposed adaptation of Brown clustering does not change the complexity of the algorithm, and—although we require that syntactically parsed text is available—we find out that much less data is needed for a desired level of clustering quality.

6.2 The Brown et al. (1992) clustering

The Brown clustering (Brown et al., 1992) is an agglomerative (merge-based) algorithm that induces a hierarchical clustering of words. It takes a tokenized corpus and groups word types into k clusters identified with unique bit strings, representing paths in the induced binary tree in which the leaves are word clusters. Prefixes of the paths can be used to achieve clusters of coarser granularity (Sun et al., 2011; Turian et al., 2010). The obtained clusters contain words that are semantically related, or are paradigmatic or orthographic variants. We are using the term *semantic relatedness* in its broadest possible scope. We say that words or clusters are semantically related if they are in any kind of semantic relation: synonymy, meronymy, antonymy, hypernymy etc. (Turney and Pantel, 2010). Figure 6.1 illustrates the clustering output as a binary tree with the bit string identifiers.

The algorithm consists of two phases. In the first phase, the algorithm starts by putting the k most frequent word types into distinct empty clusters. Then, the k+1$^{\text{th}}$ most frequent word is assigned to a new cluster, after which two among the resulting k+1 clusters are merged, namely the pair that maximizes the average mutual information of the current clustering. This process is repeated until all word types have been merged. In the second phase, the resulting k clusters are merged to build the binary tree. The version of the algorithm optimized for speed runs in $O(k^2|V|)$, where $|V|$

Figure 6.1: An illustration of a clustering tree hierarchy.

is the vocabulary size.

Brown clustering has been used extensively in NLP tasks such as parsing (Koo et al., 2008; Candito and Crabbé, 2009; Haffari et al., 2011; Pitler, 2012; Kiperwasser and Goldberg, 2015), named-entity recognition (NER) and chunking (Turian et al., 2010), sentiment analysis (Popat et al., 2013), relation extraction (Plank and Moschitti, 2013), unsupervised semantic role labeling (Titov and Klementiev, 2012a), question answering (Momtazi et al., 2010), POS tagging (Owoputi et al., 2013) and speech recognition with recursive neural networks (Shi et al., 2013). Recently, multilingual clustering has been proposed (Täckström et al., 2012; Faruqui and Dyer, 2013), and alternative ways of feature extraction from Brown clusters have been investigated by Derczynski and Chester (2016).

Among the most frequently recognized limitations (cf. Koo et al. (2008), Chrupala (2011) and Frank et al. (2013)) are a) the hard clustering approach, b) the relatively long running time[1] and c) the insensitivity to wider context. Our method attempts to overcome the final disadvantage. As it requires less data to obtain a similarity result as high as that of the standard Brown clustering, it also reduces the running time.

Leveraging syntactic context for word representations has been explored, among others, by Lin (1998b) for distributional thesauri; by Haffari et al. (2011) for combining Brown clusters and word groupings from split non-terminals; by Sagae and Gordon (2009) for using unlexicalized syntactic context in hierarchical clustering; by van de Cruys (2010) and Padó and Lapata (2007) in comparing window- and syntactic-based word space models; and Boyd-Graber and Blei (2008) in the context of syntactic topic models.

The work closest to ours is that of Grave et al. (2013). The authors show that clusters obtained from dependency trees outperform standard Brown clustering when used as features in super-sense tagging and NER. Their focus, however, is on a generalization of Brown clustering with Hidden Markov models (HMMs) and on an extension of Markov chains to trees.

[1] Especially when $k > 3000$. Note that the algorithm depends quadratically on k.

Learning and inference are done with online expectation-maximization and belief propagation. In the approach of Grave et al., the mapping between a word and its semantic class is non-deterministic, and can thus capture homonymy and polysemy. At test time, a context-sensitive inference can be performed to obtain the word representations (Huang et al., 2011; Nepal and Yates, 2014). The context sensitivity is certainly an important advantage over Brown clustering in which the mapping between a word and a cluster is deterministic. However, it has its own disadvantages: Creating context-sensitive representations requires possibly costly inference; furthermore, HMM-based clustering does not build nor lends itself easily to a hierarchy, which is often exploited during feature creation in supervised learning to control cluster granularity (see section 6.5.2). We explore context-sensitive representations in our own work in the next chapter (using HMM-based representations) and Chapter 7 (using multi-sense word embeddings).

Compared to Grave et al. who focus on new representation learning methods with HMMs on dependency trees, in this chapter we take an in-depth look at the parameters and the choices that are standardly considered when using the Brown et al. (1992) algorithm. We show that the advantage of dependency clustering can be observed throughout different parametrizations of cluster capacity, granularity level, frequency thresholding and other criteria (section 6), and that the advantage is roughly constant for varying amounts of input data.

6.3 The extension with a dependency language model

The bigram language model underlying Brown clustering takes the probability of a sentence as the product of probabilities of words based on immediately preceding words. In contrast, we replace this by a *dependency* language model (DLM), which defines the probability of a sentence over a dependency tree (Shen et al., 2008). We will define a dependency tree as a rooted, connected, acyclic directed graph in which the nodes are the

words from the vocabulary V. We will refer to the lexicalized tree as S^T and the bare tree structure as T. For our purposes, it will be sufficient to deal with unlabeled trees, i.e. we discard the dependency relations associated with the edges in the tree. The sentence probability under a DLM can be factorized in different ways (Chen et al., 2012; Charniak, 2001; Popel and Mareček, 2010), but the common idea is that a word is conditioned on some history where the link between the two is obtained with dependency edges. In practice, the history often includes the immediate parent of the word—which can be either a lexical head or the artificial root node—and sometimes siblings occurring between the surface realizations of the child and the parent. Our take on the DLM follows that of Charniak (2001) and Popel and Mareček (2010), and consists of conditioning the probability of a word on its parent:

$$p(S^T|T) = \prod_{i=1}^{m} p(w_i|w_{\pi(i)}, T), \tag{6.1}$$

where $\langle w_i \rangle_{i=1}^{m}, w_i \in V$ is the word ordering corresponding to a breadth-first enumeration of the tree, $\pi : \{1, \ldots, K\} \mapsto \{0, \ldots, K\}$ identifies the single parent of a word, with w_0 representing the root of the tree.

The objective in Brown et al. (1992) is specified in the context of a *class-based* bigram language model. The intuition for using this subset of language models relies on assigning distributionally similar words to the same classes. This can allow better predictions for histories that have not been previously observed by assuming that they are similar to other, observed histories. The clustering objective is then to find such a deterministic clustering function σ mapping each word from the vocabulary to one of the clusters, $\sigma : V \mapsto C$, that maximizes the likelihood of the data.

Here, we use the DLM as defined above to replace the bigram formulation in the objective:

$$L(S^T; \sigma, T) = \prod_{i=1}^{m} p(w_i|\sigma(w_i))p(\sigma(w_i)|\sigma(w_{\pi(i)}), T), \tag{6.2}$$

where i ranges over all children in the tree and π is the parent-assigning function introduced above. The class transition probability is conditioned on the class of the parent of w_i instead of on the class of the immediately preceding word. As shown by Brown et al. (1992), the equation (6.2) can be decomposed to the entropy of the word distribution and the mutual information between parent-child clusters. Since the entropy is independent of the clustering function, the objective amounts to finding such clustering function σ that maximizes the mutual information. We include the rewriting of the equation (6.2) in terms of mutual information in Appendix A.

On the practical side, the calculation of the mutual information in the implementation changes to the extent that count tables no longer represent the adjacency relationship (bigrams) between words but the parenthood (child–parent relation).

6.4 Experimental setup for Dutch

We evaluate our word clusters by following the method of van de Cruys (2010) for evaluating vector space models. The method is based on a wordnet for Dutch and assumes that two semantically related words are also located close to each other in the wordnet hierarchy.[2] We use Cornetto (Vossen et al., 2013), which includes around 92,000 form-POS pairs described in terms of lexical units, synsets and other criteria. We refer the reader to section 5.2.3 for details. For calculating similarity scores, we use the Lin similarity measure (Lin, 1998b; Sánchez et al., 2011), ranging between 0 and 1.

Evaluation is guided by a list of 10,000 most frequent words from SoNaR Oostdijk et al. (2008), a 500M-word reference corpus for Dutch.[3] Every word is compared to all other words in the same cluster, and the average similarity for all comparisons is taken as the final score. The de-

[2]For English, several semantic similarity datasets are available, some of which can identify the type of relatedness captured (Faruqui and Dyer, 2014a). We are not aware of such datasets for Dutch.

[3]`http://lands.let.ru.nl/projects/SoNaR`

scribed method is well suited for measuring intracluster quality, yet useful information about word similarity is available also by looking at neighboring clusters in the binary tree. This *inter*cluster quality, according to which clusters that are close in the binary tree are more similar than clusters that are far apart, can be captured indirectly by evaluating with differently sized bit strings. In this way, when a short string is used, two or more semantically related, but originally isolated clusters are merged, which should result in a drop in clustering quality (semantic relatedness tends to "dissolve" when merging).

For both the standard and the dependency Brown clustering, the same set of sentences is used. To make the experiments computationally practical, we sampled from SoNaR roughly 46M-word worth of sentences, which is comparable to the count of English datasets used in Koo et al. (2008) and Turian et al. (2010). The sentence length was restricted to five or more words to exclude noisy text. All the corpus annotation was removed.

For dependency clustering, the dataset was lemmatized and parsed with Alpino (van Noord, 2006). A description of the parser can be found in section 5.2. Unless specified otherwise, we use first-order dependencies produced by the parser. The bilexical counts of head and dependent (ignoring the relation label) serve as input for dependency clustering.

The corpus domains that we encounter in test time might be different from those of the corpus on which the parser was tested. As a consequence, the parser might not always perform with about 90% accuracy reported for newspaper texts (van Noord, 2010). However, we expect the drop in accuracy to be small since it has been shown that the Alpino parser is relatively insensitive to domain shifts compared to some entirely data-driven parsers (Plank and van Noord, 2010).

6.5 Empirical study for Dutch

The main parameter in Brown clustering is the number of clusters k, which we set to either 1000 or 3200, except when measuring clustering capacity,

for which smaller values of k are used. These values of k are also standardly encountered throughout the literature. For k above 3200, the algorithm stops being practical on current hardware assuming a single-core implementation.

We limit the minimum frequency of words in clustering to three, unless stated otherwise. The resulting vocabulary size in the case of $k = 1000$ with the frequency threshold applied is around 240,000. We use a paired t-test to check for statistical significance of observed differences in mean Lin scores.

6.5.1 Cluster examples

In Table 6.1, we show both the versatility of dependency clusters by dividing the examples in five groups (A–E), and the similarity of clusters within the group. Note that the longer the common substring between the clusters, the closer they are in the binary tree. Group **A** includes words describing professions or people's roles and functions. Group **B** lists personal pronouns, including reflexive pronouns (B2), where substantial differentiation exists with many singleton clusters. Because first and last names are very common in our corpus, many fine-grained distinctions are created between these (**C**). C1 groups names of presidents, whereas C2 and C3 distinguish between feminine and masculine names. **D** includes two related clusters describing means of communication: D1 contains orthographic variants (e.g. "email" and "e-mail") and diminutives, e.g. "sms_DIM", corresponding to Dutch "smsje"; whereas D2 includes words for communication devices and software. Finally, measurable concepts are included in **E**.

Complete clusters induced with either the standard or the dependency Brown clustering algorithm can be found at `https://github.com/rug-compling/dep-brown-data`.

Group	Cluster id	Most frequent words	Left
A1	001010001011100	aannemer, huis_arts, bakker, notaris, apotheker, makelaar *contractor, family doctor, baker, lawyer, pharmacist, estate agent*	+57
A2	001010001011011	analist, criticus, waarnemer, kenner, commentator, mens_recht_organisatie *analyst, reviewer, observer, expert, commentator, human rights organization*	+8
A3	00101000101111110	ondernemer, zakenman, bedrijf_leider, zelfstandige, koopman, starter *entrepreneur, businessman, manager, self-employed, merchant, starter*	+18
B1	011101111011110	mij *me*	0
B2	01110111101110	zichzelf, mezelf, jezelf, onszelf, mijzelf, uzelf *him/herself, myself, yourself, ourselves, myself, yourself*	0
B3	01110111101101	hem *him*	0
B4	01110111101100	hen *them*	0
C1	0011001010	Bush, Obama, Clinton, Poetin, Chirac, Sarkozy *Bush, Obama, Clinton, Putin, Chirac, Sarkozy*	+95
C2	0011000111010	Sarah, Kim, Nathalie, Justine, Kirsten, Tia, Eline	+12
C3	0011000111011	David, Jimmy, Benjamin, Samuel, Tommy, Sean	+98
D1	001011100010101	email, mail, sms, sms_DIM, e-mail, mail_DIM	+13
D2	001011100010100	telefoon, satelliet, telefonie, telefoon_lijn, Explorer, muziek_speler, iTunes *telephone, satellite, telephony, telephone line, Explorer, music player, iTunes*	+7
E	001000010110101	inkomen, energie_verbruik, minimum_loon, cholesterol, *income, energy consumption, minimum wage, cholesterol,* opleidingsniveau, IQ, alcohol_gehalte *level of education, IQ, alcohol content*	+32

Table 6.1: Example dependency clusters obtained with k set to 3200 and minimum frequency to 50. The underlined parts of the bit strings indicate the longest common substring within a group. English translation of the Dutch original is given in italics. Column *Left* indicates the remaining number of (less frequent) words in the cluster.

6.5.2 Cluster quality

Table 6.2 presents the general quality of standard and dependency clustering. The results for 1000 and 3200 clusters show that we obtain a higher similarity score for 3200 clusters compared to 1000, and a more marked difference between the standard and the dependency clustering in the case of k=3200 (Δ=0.019). We also look at how many words from the frequency list were evaluated successfully. The recall depends on the success of mapping between words and synsets as well as the success of finding the word in one of the clusters. The latter factor influences the recall to a much lesser degree, as almost all words are found in the clustering. For 3200 clusters with the minimum frequency set to fifty, approximately 5000 words are successfully evaluated, whereas for 1000 clusters, this number is around 7000 due to a different frequency threshold. These numbers are not affected by the type of clustering (standard or dependency).

k	Brown	DepBrown	Δ
1000	0.191	0.196	$+$.005*
3200	0.279	0.298	$+$.019**

Table 6.2: Lin similarity scores for the standard (*Brown*) and the dependency Brown clustering (*DepBrown*), with k the number of clusters. Δ=DepBrown – Brown. Frequency threshold of 50 is used for clustering with $k = 3200$. *: statistically significant with $p < 0.05$, **: statistically significant with $p < 0.001$.

The results under three different clustering parametrizations are shown in Table 6.3. One way of controlling the granularity is to choose the number of output clusters k. As shown in the table under CAP ("capacity"), the dependency clustering achieves a better quality regardless of the choice of k, and in general, choosing a smaller k decreases the quality, which is compatible with the observations of Turian et al. (2010) in their chunking experiments.

It is often the case that frequent words within a single cluster exhibit

Setting	k	*min*	Brown	DepBrown	Δ
CAP	200	10	0.148	0.157	+.009
	400	10	0.169	0.175	+.006
	600	10	0.182	0.191	+.009
	800	10	0.191	0.205	+.014
FREQ	1000	5	0.196	0.204	+.008
	1000	10	0.202	0.216	+.014
	1000	20	0.206	0.221	+.015
	1000	30	0.209	0.224	+.015
	1000	50	0.216	0.227	+.011
NOUNS	1000	3	0.272	0.279	+.007

Table 6.3: Lin similarity scores for the standard *Brown* and the dependency Brown clustering (*DepBrown*), with k the number of clusters, min the minimum frequency of words. CAP: varying k, fixed min; FREQ: varying min, fixed k; NOUNS: evaluating only nouns, Δ=DepBrown – Brown. All the results reported for *DepBrown* are significantly different from *Brown* with $p < 0.001$.

clear semantic relatedness, but rare words are semantically quite unrelated.[4] This is confirmed by our results in which the quality of the clustering improves approximately logarithmically with frequency threshold increasing (FREQ). The margin between the standard and the dependency clustering is also increasing as we increase the threshold. In downstream tasks, Brown clusters bring improvements with both a high frequency threshold (Owoputi et al., 2013) as without thresholding (Koo et al., 2008; Turian et al., 2010).

We also investigate the quality of nouns only to facilitate the comparison to van de Cruys (2010). We observe a considerable gain in quality when only nouns are used compared to using all parts of speech—the Lin score is increased by 0.08. In the noun-only evaluation, the dependency clustering also achieves a higher score (0.279) than the standard clustering (0.272). It

[4] Although cf. Turian et al. (2010) who show that Brown clustering has a superior representation for rare words than the neural word embeddings in their experiment.

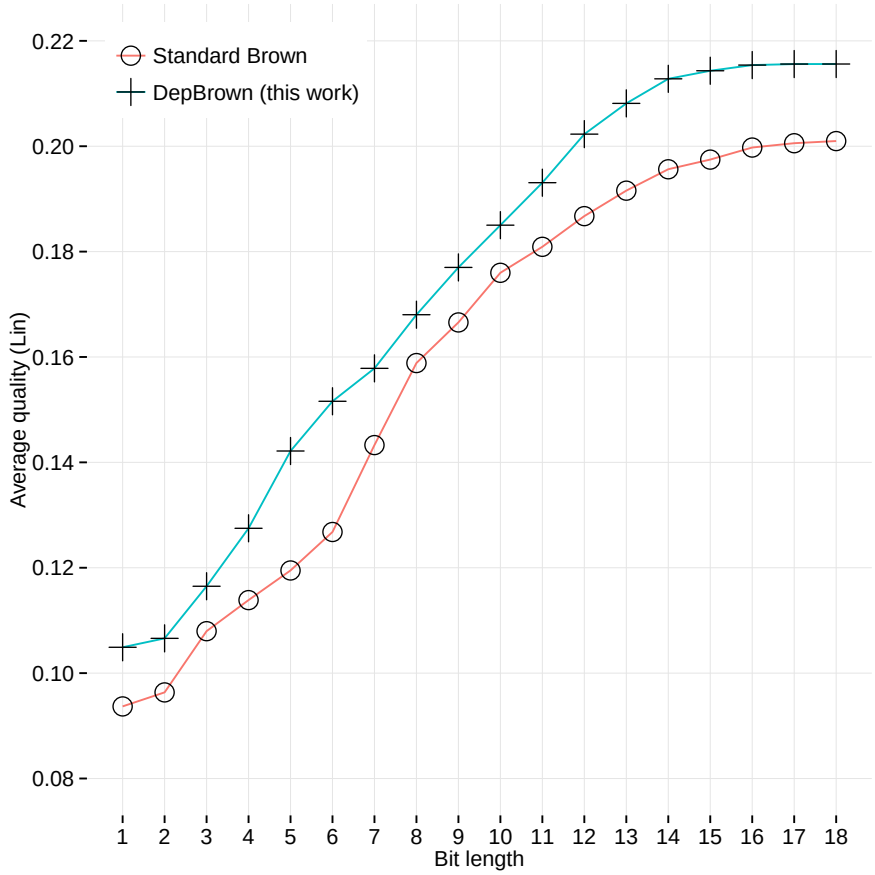


Figure 6.2: Effect of bit string length of the standard and the dependency Brown clustering on the similarity scores. The number of clusters parameter set to 1000, and frequency threshold to 10.

is shown in van de Cruys (2010) that syntactic vector space models outperform window-based models, which is confirmed by our finding for word clustering as well. In his work, the syntactic vector space models yield a 0.04 advantage in Lin score, whereas our dependency clusters achieve a

less marked advantage, reaching up to 0.019 in Lin score. A possible explanation for this difference is that in his evaluation an average over only five most similar nouns is taken, whereas we impose no such restriction. We would like to point out that our work does not aim to compare and discuss the merits of clustering and vector space models as two competing techniques for obtaining word representations, but rather to provide a comprehensive comparison of the standard Brown clustering to its dependency extension.

An effect similar to that of controlling capacity with the parameter k can be achieved by making use of the fact that the induced structure is a hierarchy. While the parameter k needs to be chosen before clustering, the hierarchical structure can be exploited during feature extraction based on the already existing clusters. By truncating the path bit string, we can control the cluster granularity. For different tasks, different bit string lengths might be appropriate (Sun et al., 2011; Koo et al., 2008; Miller et al., 2004). For example, one might prefer coarser distinctions (i.e. shorter bit strings) in parsing, while finer granularity might be necessary to obtain effective representations of proper names in NER. We have ran the experiment with string length ranging from one to eighteen, and show the results in Figure 6.2. Across the board, the dependency clustering yields better results than the standard clustering. Naturally, with shorter strings the quality decreases, which is explained by an increasing word population in the clusters, with more and more distant (both hierarchically and semantically) clusters being merged.

6.5.3 Amount of data

Figure 6.3 shows the amount of data needed to achieve a certain quality of clustering. For clustering on one hundred thousand sentences the similarity score for DepBrown is around 0.165 and for Brown around 0.155. For each subsequent addition of the data, the dependency clustering continues to outperform the standard clustering, with a single exception at 2.2 million sentences. In order to achieve the highest score attained by the

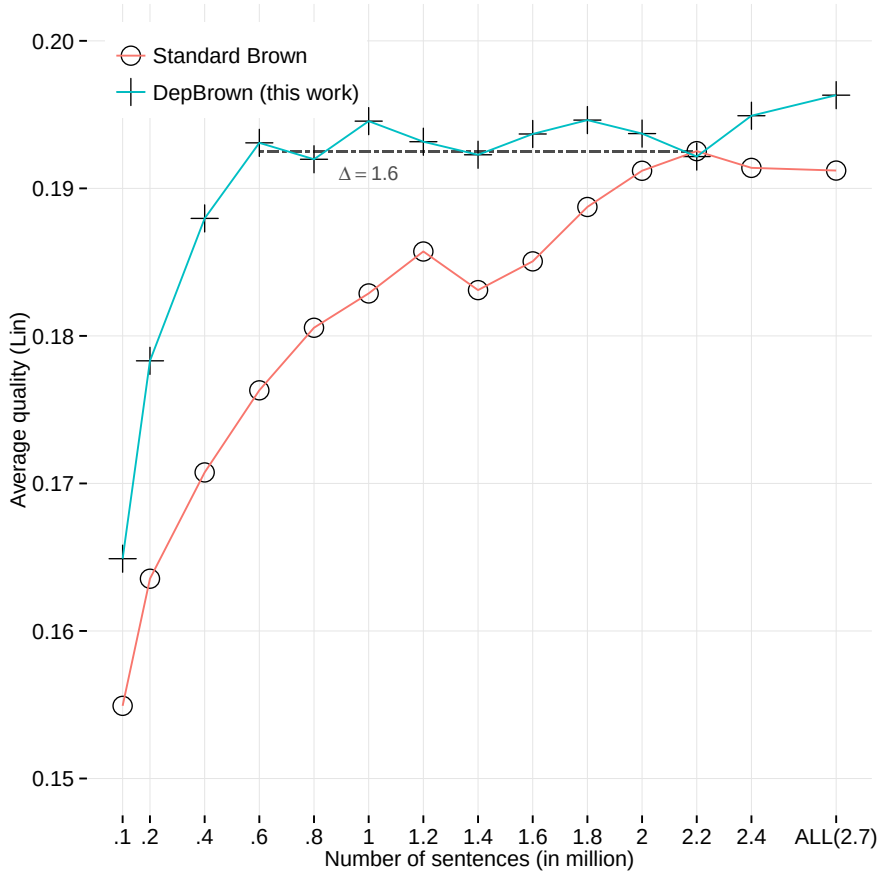


Figure 6.3: Learning curves for the standard and the dependency Brown clustering with 1000 clusters. Dashed line displays the difference in amount of data (in millions of sentences) needed for *DepBrown* to achieve the best quality of *Brown*. Using all 2.7 million sentences from the corpus corresponds to 46 million words.

standard clustering (0.193) with 2.2 million sentences, dependency clustering requires only around 0.6 million sentences, which is around 1.6 million

sentences (or 27 million words) less. This observation is advantageous especially because less data means shorter running time for clustering as the number of word types is reduced.[5]

6.5.4 Data selection with dependency relations

Our dependency clustering described in the previous sections operates on words appearing in any dependency relation. We now investigate whether selecting only a particular dependency relation—i.e. using both parent and child words from that dependency relation as input—leads to clusters with stronger semantic relatedness. Each relation can be characterized as either a first- or a second-order relation. A second-order relation is between the words with an intervening preposition, e.g. between a verb and a noun of a directional complement introduced by a preposition, such as in the Dutch "eten achter pc" ("eating at the computer").[6] We have induced distinct sets of clusters for each of the forty-five dependency relations and measured the quality of each of them separately. The cumulative baseline that does not distinguish between dependency relations (studied before) is given in Table 6.4 as ALL (Ord-1). This is the same result as reported in Table 6.2 for $k = 1000$. The addition of second-order dependencies (Ord-1 & Ord-2) does not change the clustering quality of the baseline (0.196) but increases the number of types.

In the upper part of Table 6.4, we list six relations that lead to a clustering quality which is above the baseline. Two conclusions can be drawn from the results on these relations. First, some dependency relations contribute better context that leads to increased semantic relatedness compared to clustering without relation selection. Second, both the first- and the second-order relations appear among the relations outperforming the baseline. The highest score from the top six relations is achieved by using

[5]Of course, in practice the advantage is only applicable when the parsed text already exists.

[6]The preposition should be seen only as an implicit link between the words and is not included in the input data for clustering. For the example fragment, "eating-computer" would constitute a data instance used by the algorithm.

Type	Ord-1	Ord-2	DepBrown	Population
OBJ2		■	0.238	1,622
LD	■		0.233	2,419
PC		■	0.211	21,157
LD		■	0.208	12,149
OBJ1	■		0.203	108,037
SU	■		0.199	79,844
ALL	■		0.196	495,479
ALL	■	■	0.196	559,908
SU+OBJ1	■		0.202	156,645

Table 6.4: Lin similarity scores for the dependency Brown clustering (*DepBrown*) per type of dependency relation. Ord-1: first-order relation; Ord-2: second-order relation (with intervening preposition); Population: number of word types in the clustering.

the instances exclusively from the second-order secondary object (OBJ2) relation. However, relatively few word types are included. The same is true for the first-order directional complements (LD). Of course, clustering with only one of these relations would have quite limited applicability if used in a downstream NLP task due to the low number of word types. However, the main point we want to make is that these relations yield semantically superior clusters, and as such deserve further attention in learning semantic clusters using syntax. The remaining four among the top six relations are more frequent, and lead to sets of clusters with higher number of word types. These are the second-order prepositional complement (PC) and directional complement (LD), and the first-order direct object (OBJ1) and subject (SU) relations. Finally, the setting SU+OBJ1 joins words obtained from the two largest categories, the subject and the direct object relations. It achieves a quality that falls somewhere between the values obtained for the two relations separately, and still increases the number of word types.

6.6 A study on English

We now present another empirical study to investigate whether dependency clustering in English leads to the same advantage as observed in Dutch. Throughout, we follow the same experimental setup as for Dutch, but the resources and the syntactic parser are different. We now describe those in more detail, and then present a summary of our findings.

6.6.1 Experimental setup

We carry out the wordnet similarity evaluation task in the same way as described in section 6.4, however with a wordnet for English, namely WordNet v.3 (Miller, 1995), conveniently included in the NLTK library (Bird et al., 2009). We use the implementation of the Lin similarity measure from NLTK, which differs from the one we have used for Dutch in that it is based on an information-theoretic notion of information content[7], and amounts to taking the inverse of the frequency of a word in a corpus. We use the IC scores already available in NLTK, precomputed from the British National Corpus Leech (1992) for nouns and for verbs.

We use the BLLIP (Charniak et al., 2000) corpus to induce the clusters for English. The corpus contains around 43 million words of Wall Street Journal text. We ensure that the sentences of the Penn Treebank, which is used to train our dependency parser, are removed prior to clustering. Although this step is only really needed for dependency clustering, we use the filtered corpus for both types of clustering to enable a fair comparison. To obtain the dependency tuples that serve as input for dependency clustering, we parse the corpus with the MST parser (McDonald and Pereira, 2006). We use the projective, second order model variant, and train it on sections 2–21 of the Penn Treebank Wall Street Journal (PTB). Prior to that, the PTB was patched with NP bracketing rules (Vadas and Curran, 2007) and converted to dependencies with the LTH converter (Johansson and Nugues, 2007). Our trained parsing model achieves the unlabeled accuracy

[7]The IC criterion for Dutch was defined ontologically, see equation (5.3).

of 91.84 and the labeled accuracy of 87.36 on section 23 of the PTB without retagging the POS. At train and test time, the parser relies on POS tags. In our case, we assign these with Citar (`https://github.com/danieldk/citar`), a trigram HMM POS tagger inspired by the work of Brants (2000). We use the standard splits of the PTB for training and testing, sections 00–18 and 22–24, respectively. The tagger achieves the accuracy of 96.3. The output of the parser is a CoNLL-formatted file, from which we extract dependency tuples of head–dependent pairs in any dependency relation. The vocabulary size for the base setting of clustering with the minimum frequency threshold of 3 is around 320,000 for English. This number is somewhat higher than in Dutch due to the use of word forms in English and word roots in Dutch.

6.6.2 Empirical summary

Setting	k	*min*	Brown	DepBrown	Δ
CAP	200	3	0.172	0.168	−.004
	400	3	0.187	0.181	−.006
	600	3	0.199	0.190	−.009
	800	3	0.204	0.197	−.007
	3200	50	0.319	0.313	−.006
FREQ	1000	3	0.212	0.201	−.009
	1000	5	0.225	0.212	−.013
	1000	10	0.240	0.229	−.011
	1000	20	0.255	0.246	−.009
	1000	30	0.264	0.256	−.008
	1000	50	0.276	0.266	−.010
NOUNS	1000	3	0.215	0.206	−.009

Table 6.5: Lin similarity scores for the standard *Brown* and the dependency Brown clustering (*DepBrown*) for English, with k the number of clusters, min the minimum frequency of words, Δ=DepBrown – Brown.

The similarity scores for English are reported in Table 6.5. Under all

investigated conditions—varying the clustering capacity, the frequency of words for inclusion and the restriction to nouns, the situation is the reverse of what we have seen for Dutch, with dependency Brown clusters failing to outperform the standard clusters. Another difference to the situation for Dutch is that here we only observe a very modest increase in the quality when constraining the POS tags to nouns (as opposed to using all POS). One explanation for this might be that in English the evaluation covers only nouns and verbs, while in Dutch it also includes adjectives and adverbs. For these, as well as for verbs, the quality in Dutch is lower than for nouns.

How can we interpret the finding that the the dependency representation in Brown clustering for English has such a different effect than for Dutch? One possible explanation is that the input tuples are of different quality or type. For example, the dependency parser for English might make more mistakes than the parser for Dutch. However, we expect that this effect should be small, if at all existing, since the parsing accuracies measured on the test sets are comparable in both Dutch and English. It is also unlikely that the accuracy of the English parser could be affected by a domain shift. The BLLIP corpus and WSJ have very similar domains, chiefly economical news. In addition, the dependency parsing schemes might follow different attachment conventions, leading to extraction of a different types of tuples, or contexts. An example is the case of coordination attachment, for which the convention is to link the coordinated elements together in Dutch, but not in English. [8]

Next, it might be the case that the effect can be accounted by the differences between languages (Bender, 2011). The word order is generally accepted to be less fixed in Dutch than in English. This means that the standard bigram clustering approach can capture the selectional preferences less well than in English. Because of that, the dependency representation could be more useful for Dutch than for English.

[8] In the sentence "Enkele belangrijke en actieve leden (naast Vlaanderen) zijn [...]" ("Some important and active members (next to Flanders) are ..."), a conjunction relation would couple "belangrijke" and "actieve", which would consequently constitute a tuple.

Lastly, the effect could be related to the specific task. However, given the findings from the subsequent study on HMM-based representations (Chapter 7), this factor is unlikely to be strong as the same language pattern emerges there.

6.7 Additional related work

The topic of linear versus syntactic context in word cluster induction has been studied for English by Grave et al. (2013), which is the work most related to ours. In their research, the semantic clusters are seen as a special case of HMM-induced representations in which the model's function for mapping words to clusters is deterministic. The main focus of Grave et al. is on continuous, non-deterministic HMM representations, and on the techniques for efficient learning. While the Brown clusters, both standard and dependency-based, are used in their work, they only include the clusters in default form without exploring the clustering parameters as we do here. The role of context type in Brown clustering has been first studied by Momtazi et al. (2010), but in a task-specific setting that includes term extraction for sentence retrieval. They compare differently sized linear contexts, window-2 and window-5, as well as a sentence- and a document-based context type. In addition, they include dependency contexts obtained by parsing with Minipar (Lin, 2003). They find that bigram-based clusters outperform all other context types when evaluated in sentence retrieval.

A substantial amount of work on the properties of linear and syntactic contexts has been done in the domains of automatic detection of similar words (Lin, 1998a; Grefenstette, 1994; Ruge, 1992; Hindle, 1990) and in distributional word-space modeling (Padó and Lapata, 2007; Baroni and Lenci, 2010; van de Cruys, 2010) as well as in topic modeling (Boyd-Graber and Blei, 2008). In the framework of distributed word representations (word embeddings), the contribution of dependency contexts has been explored by Levy and Goldberg (2014a). While all these methods ultimately try to capture some broad notion of semantic similarity, they differ

in how they represent words: Word space models use high-dimensional vectors of sparse co-occurrence counts; distributed word representations as well as HMM-based word representations use lower-dimensional (typically around 100) dense vectors; and Brown clusters use discrete binary codes to uniquely identify the word cluster in the hierarchy.

6.8 Conclusion

We have presented a detailed study on extending the Brown clustering algorithm with a dependency language model. In the first part, we have shown the advantage of the dependency clustering over the standard Brown algorithm in a series of experiments in which we have investigated the impact of cluster capacity, granularity level, frequency thresholding, amount of data and more. In the second part, we put forward the idea of selective clustering using data obtained only from specific dependency relations. Several relations lead to an improved intracluster similarity.

Our findings from the experiments on selective clustering warrant the development of more complex models that could use the additional information available in the form of syntactic functions in the creation of semantic clusters. This is the topic of the next chapter.

In the present chapter, we have analyzed the cluster quality by means of a wordnet similarity task. We have chosen this method because it is informative about the generic quality of the clusters, i.e. one that is not linked to any particular task. Still, we need to acknowledge that the relationship and the qualitative difference between the dependency and the standard clustering might not be the same in a downstream NLP application. Although the dependency clusters will not be the focus of the next chapter, we will still use them as baseline features, and will report also on their performance in extrinsic tasks, for both Dutch and English.

Another interesting question which we do not address in this thesis is how the parsing accuracy affects the quality of obtained clusters.

CHAPTER 7

A discrete latent-variable model with syntactic functions

In this chapter, we build on the work from the previous chapter and consider a model which will not only be based on syntactic structure, but will also use the labels of syntactic relations. Unlike in Brown clustering in which only hard assignments to semantic classes are possible, the latent-variable models discussed here can provide distinct word representations depending on the context in which words occur. Since word representations induced from models with discrete latent variables such as Hidden Markov models (HMMs) have been shown to be beneficial in many NLP applications, we use HMMs as the basis for our work. We exploit labeled syntactic dependency trees and formalize the induction problem as unsupervised learning of tree-structured hidden Markov models. Syntactic functions are used as additional observed variables in the model, influencing both transition and emission components. Such syntactic information can potentially capture more fine-grained and functional distinctions between words, which, in turn, may be desirable in many NLP applications.

We evaluate the word representations on two tasks—named entity recognition and semantic frame identification. Compared to an unlabeled tree model, we observe improvements from exploiting syntactic function information in both applications, and find that the obtained results rival those of state-of-the-art representation learning methods. Additionally, we revisit the relationship between sequential and unlabeled-tree models and find that the advantage of the latter is not self-evident.

7.1 Hidden Markov models and their variants

Although the methods for obtaining word representations are diverse (see previous chapters), they normally share the well-known distributional hypothesis (Harris, 1954), according to which the similarity is established based on occurrence in similar contexts. However, word representation methods frequently differ in how they operationalize the definition of context. We have discussed in the previous section how syntactic contexts can lead to superior representations compared to those obtained from linear sequences in several downstream tasks (Grave et al., 2013; Bansal et al., 2014; Sagae and Gordon, 2009; Belinkov et al., 2014) as well as when evaluated on semantic similarity tasks (Levy and Goldberg, 2014a; Padó and Lapata, 2007).

Unlike the majority of recent work pursued in the context of neural network-inspired word embeddings (Collobert and Weston, 2008; Mikolov et al., 2013a; Pennington et al., 2014), the methods discussed in this chapter fall into the framework of hidden Markov models, building on the previous work of Grave et al. (2013) and Huang et al. (2014). Although HMMs are much less explored for the purpose of word representation learning, they are appealing due to their ability to provide context-sensitive representations, in the sense that the same word in two different sentential contexts can be given distinct representations. In this way, we are able to account for various senses of a word. While this advantage comes naturally in HMMs, the handling of polysemy and homonymy typically requires a substantial

model extension in other frameworks, cf. Huang et al. (2012), Tian et al. (2014), Neelakantan et al. (2014) as well as our own work described in the next chapter. On the downside, context sensitivity in HMMs requires inference, which is expensive compared to a simple lookup, so we concentrate in our experiments on word representations that are originally obtained in a context-sensitive way, but are available for lookup as static representations at test time.

In our method, we include two types of observed variables, namely words and syntactic functions, and a latent variable which captures the meaning of the word. The added observed syntactic function variable allows us to address a drawback of learning word representation from unlabeled dependency trees in the context of HMMs (section 7.2). The motivation for including syntactic functions comes from the intuition that they in fact act as proxies for semantic roles. The current research practice is to either discard this type of information (so context words are determined on the syntactic structure alone (Grave et al., 2013)), or include it in a preprocessing step, i.e. by attaching syntactic labels to words, as in Levy and Goldberg (2014a).

We evaluate the word representations in two structured prediction tasks, named entity recognition and semantic frame identification. As our extension builds upon sequential and unlabeled-tree HMMs, we also revisit the basic difference between these two architectures, but are unable to confirm entirely the alleged advantage of unlabeled syntactic context for word representations in the named-entity recognition task.

7.2 Motivating syntactic functions

A word can typically occur with distinct syntactic functions. Since these account for words in different semantic roles (Bender, 2013; Levin, 1993), the incorporation of the syntactic function between a word and its parent[1]

[1]Here, just like in the previous chapter, we adhere to the "parent"–"child" terminology, although "governor"/"head"–"dependent"/"child"/"modifier" are also found in the literature (Kübler et al., 2009).

could give us more precise representations. For example, in "Carla bought the computer", the subject and the object represent two different semantic roles, namely the buyer and the goods, respectively. Along similar lines, Padó and Lapata (2007), Séaghdha and Korhonen (2014) and Grave et al. (2013) argue that it is inaccurate to treat all context words as equal contributors to a word's meaning. We have also provided some evidence in the previous chapter, in which we have seen how a data selection technique based on dependency relations affects the final semantic similarity scores.

In HMM learning, the model parameters learned from unlabeled syntactic structure encode the probabilistic relationship between the hidden states of parent and child, and that between the hidden state and the observed word. The tree structure thus only defines the word's context, but is oblivious of the relationship between the words. For example, Grave et al. (2013) acknowledge precisely this limitation of their unlabeled-tree representations by providing as example a hidden state of a verb that cannot discriminate between left (e.g. subject) and right (e.g. object) neighbors because of shared transition parameters. This adversely affects the accuracy of their super-sense tagger for English. Similarly, we have demonstrated in Chapter 6 that filtering dependency instances based on syntactic functions can positively affect the quality of obtained Brown word clusters when measured in a wordnet similarity task.

7.3 A tree model with syntactic functions

We now introduce some notation that we will use to define the tree model with syntactic functions. We represent a sentence as a tuple of K words, $\mathbf{w} = (w_1, \ldots, w_K)$, where each $w_k \in \{1, \ldots, |V|\}$ is an integer representing a word in the vocabulary V. In the vanilla inference procedure, the goal is to infer a tuple of K states $\mathbf{c} = (c_1, \ldots, c_K)$, where each $c_k \in \{1, \ldots, N\}$ is an integer representing a semantic class of w_k[2], and N is the number of

[2]In the previous chapter, the semantic class of w_k was assigned by a clustering function σ, while here we use the variable c for convenience.

Figure 7.1: Hidden Markov tree model with syntactic functions, r, as an additional observed layer. The illustration is a Bayesian network, with observed variables depicted in gray and the latent variable in white. The edges between the latent nodes represent the dependency links.

states, which needs to be set prior to training. In the continuation, we will see other, more information-rich methods by letting w_k's representation be a probability distribution over N states. In that case, we denote w_k's representation as $\mathbf{u}_k \in \mathbb{R}^N$.

As usual in Markov models, we can reason about the model architecture from the point of view of generation. The generation of the sentence in an unlabeled-tree HMM can be decomposed into the generation of classes (transitions) and the generation of words (emissions). The process is defined on a tree, in which a node c_k is generated by its single parent $c_{\pi(k)}$, where $\pi : \{1, \ldots, K\} \mapsto \{0, \ldots, K\}$, with 0 representing the root of the tree (the only node not emitting a word). We denote a syntactic function as $r \in \{r_1, \ldots, r_S\}$, where S is the total number of syntactic function types produced by the syntactic parser. We encode the syntactic function at position k as $r_k \triangleq r_{w_k \to w_{\pi(k)}}$, i.e. the dependency relation between w_k and its parent.

We would like the variable r to modulate the transition and emission processes. We achieve this by drawing on a special type of HMM, called the Input-output HMM. It was first introduced by Bengio and Frasconi (1996) as a sequential model in which an additional sequence of observations called "input" becomes part of the model, and the model is used as a conditional predictor. The authors describe the application of their model in speech processing, where the goal is to obtain an accurate predictor of the output phoneme layer from the input acoustic layer. Our focus is, in contrast, on representation learning (the hidden layer) rather than prediction (the output layer). We also adapt the sequential topology used by Bengio and Frasconi to trees.

We condition the probability distribution of words and semantic classes on syntactic functions, using the following factorization:

$$p(\mathbf{w}, \mathbf{c}|\mathbf{r}) = \prod_{k=1}^{K} p(w_k|c_k, r_k) p(c_k|c_{\pi(k)}, r_k), \tag{7.1}$$

where r_k encodes additional information about w_k, in our case the syntactic

function of w_k to its parent. This is represented graphically in Figure 7.1.

The parameters of the model are stored in column-stochastic[3] transition and emission matrices[4]:

$\mathbf{T}$, where $T_{ijl} = p(c_k{=}i \mid c_{\pi(k)}{=}j, r_k{=}l)$

$\mathbf{O}$, where $O_{ijl} = p(w_k{=}i \mid c_k{=}j, r_k{=}l)$

The number of required parameters for representing the transitions is $O(N^2\,S)$, and for representing the emissions $O(N\,|V|\,S)$. In practice, the number of syntactic functions S is small. Additionally, we find it beneficial to restrict S to most frequent syntactic functions (see section 7.1 for details).

Our model satisfies the single-parent constraint and can be applied to proper trees only. It is in principle possible to extend the base representation for the model by using approximate inference techniques that work on graphs (Murphy, 2012, section 20.4), but we do not explore this possibility in this thesis.[5]

As opposed to an unlabeled-tree HMM, which is a representative of a homogeneous model, our extension can in fact be categorized as an *in*homogeneous (or non-stationary) model since the transition and emission probability distributions change as a function of input (cf. Bengio (1999) and Murphy (2012)). Another comparison concerns the learning of long-term dependencies: Since in the Input-output architecture the transition probabilities can change as a function of input at each k, they can be sharper (have lower entropy) than the transition probabilities of a homogeneous HMM. Having the transition parameters closer to zero or one reduces the ambiguity of the next state and allows the context to flow more easily. To verify and illustrate that with a concrete example, we have used the transition parameters of two of our trained models as input data for

[3]Which means that each column sums to 1: $\forall l \in r, \forall j \in c : \sum_i T_{ij}^{(l)} = 1$.

[4]We are abusing the terminology slightly, as these are in fact three-dimensional arrays.

[5]This would be relevant for dependency annotation schemes which include secondary edges.

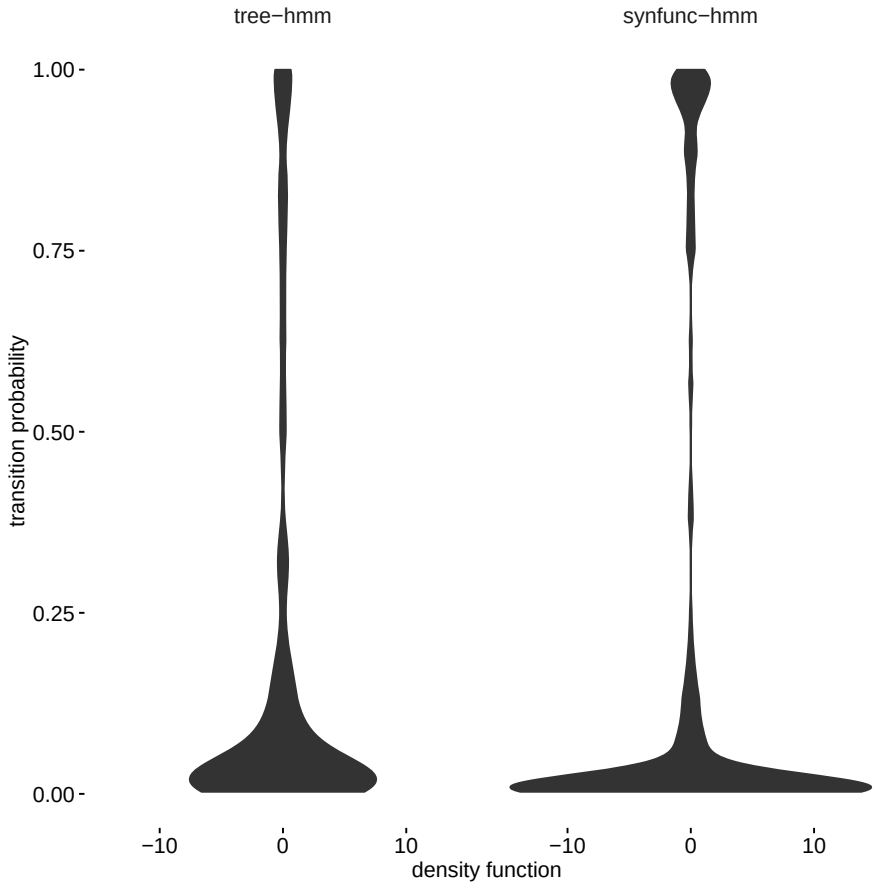


Figure 7.2: The shape of the transition probability distribution for each of *synfunc-hmm*, the tree HMM with syntactic functions, and *tree-hmm*, the unlabeled-tree HMM.

Figure 7.2. In these plots, the probability densities are shown using the width of the plot, and the continuous y-axis represents transition probability values. The violin plots make it easy to spot the relative differences in the transition probabilities of the two models. We see that in both cases,

the probability of most transitions is low, but the transition probabilities of the tree HMM with syntactic functions are more peaky than those of an unlabeled-tree HMM: The density plot is even more concentrated at the extremes, i.e. closer to either 0 or 1, whereas the middle spectrum is thinner. We confirm this difference also numerically by calculating the cumulative entropy of the transition probability distribution, which gives a lower entropy of 5.34 to the inhomogenous *synfunc-hmm* and 5.6 to the homogenous *tree-hmm* model. We can thus confirm the reasoning above about the sharper transition probability distribution in the case of inhomogenous models.

7.4 Learning and inference

We train the model with the Expectation-Maximization (EM) algorithm (Baum, 1972; Dempster et al., 1977) and use the sum-product message passing for inference on trees (Pearl, 1988). The message passing, sometimes also called belief propagation, is an instance of exact inference algorithms for graphical models, and corresponds to the forward-backward recursions (Stratonovich, 1960; Rabiner, 1989) for inference in sequential HMMs. The estimation of hidden states in our case is the same as in an unlabeled-tree model, except that it is performed conditionally on r. We provide the details of the inference algorithm in Appendix B.

The E-step uses the current model parameters to calculate, for each training example, the posterior probability of each label. In supervised learning, we need an empirical distribution $\hat{p}(X, Y)$, but in unsupervised learning, we're given only a distribution over observations, $\hat{p}(X)$. The role of the E-step is to provide an estimate of conditional probability of Y, which is not observed, given X, allowing us to obtain a complete distribution $\hat{p}(X, Y)$. The M-step then simply performs supervised learning, finding maximum-likelihood estimate (MLE) from the multinomial model family $\mathcal{P}$ given the complete sample.

In our case, we are concerned with estimating the parameters $\mathbf{T}$ and

$$\tau_{ijl} = \sum_{n=1}^{N} \sum_{k=1}^{K_n} \mathbb{E}\Big[1\{C_k^{(n)}{=}i, C_{\pi(k)}^{(n)}{=}j, R_k^{(n)}{=}l\} \;\Big|\; W^{(n)}{=}\mathbf{w}^{(n)}, R^{(n)}{=}\mathbf{r}^{(n)}\Big]$$

$$\omega_{ijl} = \sum_{n=1}^{N} \sum_{k=1}^{K_n} \mathbb{E}\Big[1\{W_k^{(n)}{=}i, C_{\pi(k)}^{(n)}{=}j, R_k^{(n)}{=}l\} \;\Big|\; W^{(n)}{=}\mathbf{w}^{(n)}, R^{(n)}{=}\mathbf{r}^{(n)}\Big]$$

Figure 7.3: Obtaining pseudo-counts, or expected sufficient statistics, in the E-step.

O of our tree HMM with syntactic functions. In the E-phase, we obtain pseudo-counts, or the expected sufficient statistics, on the basis of the existing parameters, as shown in Figure 7.3. These are calculated with the sum-product message passing algorithm and represent the number of times that each hidden variable is expected to be used with the current setting of the parameters. The M-step then simply normalizes the pseudo-counts

$$\hat{T}_{ijl} = \frac{\tau_{ijl}}{\sum_{j'} \tau_{ij'l}} \tag{7.2}$$

in the case of transition parameters and, similarly, for emissions. The complexity of the message passing algorithm is proportional to the number of nodes in the tree and the square of the number of states in the model (if a word can take on any semantic class).

7.4.1 State splitting and merging

We explore the idea of introducing complexity, i.e. the desired number of states, progressively in order to alleviate the problem of EM finding a poor solution. Note that this can be particularly severe when the search space is large (Petrov et al., 2006). The splitting procedure starts with initializing the model parameters with a small number of states, then splits the parameters of each state s into s_1 and s_2 by cloning s and slightly perturbing to

break the symmetry. The model is retrained, and a new split round takes place. The split-and-train procedure is repeated until the desired number of states is reached. To allow splitting states to various degrees—so that some states would ultimately become more fragmented than others—Petrov et al. also merge back those already split states which improve the likelihood the least. The motivation for this step is that excessive fragmentation of hidden states can yield less robust state estimates, leading to overfitting. This can be justifiable in cases when a class is relatively general and would not profit from further subcategorization.[6] Although the merge step is done approximately and does not require new cycles of inference, we find that the extra running time does not justify the sporadic improvements we empirically observe. We settle therefore on the splitting-only regime, and instead of training the model with 2^k states directly, we find it most beneficial to begin with $k-2$ or $k-1$. So, for a model in which the desired number of states is 128 ($k=7$), the splitting procedure works well when beginning with 32 ($k=5$) or 64 ($k=6$) states, whereas the performance with smaller k (<5) is actually worse than when training without splitting.

7.4.2 Decoding for HMM-based models

Once a model is trained, we can search for the most probable states given observed data by using the max-product message passing (Max-Product, a generalization of the Viterbi algorithm that works on sequential data) for efficient decoding on trees:

$$\hat{\mathbf{c}} = \arg\max_{\mathbf{c}} p(C = \mathbf{c} \mid W = \mathbf{w}, R = \mathbf{r}). \tag{7.3}$$

We have also tried posterior (or minimum risk) decoding (Lember and Koloydenko, 2014; Ganchev et al., 2008), in which we calculate the states that are individually the most probable given the observation. The method works by performing message passing to obtain the state posteriors, after

[6] E.g., a part-of-speech category for function words, like determiners (Petrov, 2009).

which the maximum scoring states can be determined. In our empirical study on the development set of the NER data, however, we find no consistent improvement over the max-product message passing.

It is also possible to omit the search for the best states and simply use the posterior state distribution $\mathbf{u}_k$ over N hidden states, obtained with the inference algorithm (Nepal and Yates, 2014; Grave et al., 2014):

$$u_c^{(k)} = \mathbb{E}\big[1\{C_k = c\} \mid W = \mathbf{w}, R = \mathbf{r}\big]. \tag{7.4}$$

We call this representation Post-Token. Since it is vectorial, it can bear more information than the optimal discrete states. It is worth emphasizing that in *both* Max-Product and Post-Token, inference is performed based on a given sentence, thus providing a context-dependent representation. We find in our experiments that Post-Token consistently outperforms Max-Product due to its ability to carry more information and uncertainty. This can then be exploited by the downstream task predictor. Similar observations have been made by Huang et al. (2014) and Nepal and Yates (2014).

One disadvantage of using context-sensitive representations at test time is that obtaining them is relatively costly. That is both because syntactic parsing is required by our models and because the computation needed for inference is more time consuming than when using a simple lookup. Another disadvantage with techniques involving inference and decoding is that these are sometimes not applicable. An example is information retrieval, where the entire sentence is usually not given (Huang et al., 2011), and only a word or two are provided as a query. Based on these arguments, it can be beneficial therefore to accept a trade-off between full context sensitivity and efficiency; this can be achieved in our case with an alternative, static representation called Post-Type. These representations can be obtained in a context-insensitive way at test time (Huang et al., 2011; Grave et al., 2014). In order to prepare them, we simply average the posterior state distributions (which are context-sensitive) of *all* occurrences of a

word type $\tilde{w}$ from a large corpus U:

$$\mathbf{v}^{(\tilde{w})} = \frac{1}{Z_{\tilde{w}}} \sum_{i \in U : w_i = \tilde{w}} \mathbf{u}^{(i)}. \tag{7.5}$$

In other words, we obtain a generic, one-vector-per-word representation, in which the different senses found in different contexts are in fact conflated.

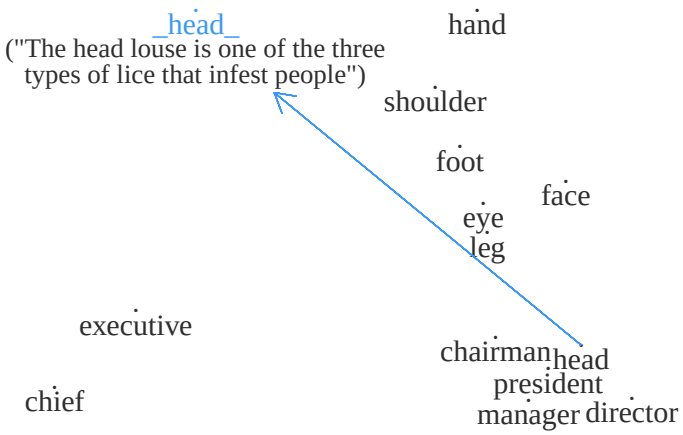


Figure 7.4: Example representations obtained with our model with syntactic functions. All are static Post-Type representations, except "_head_", which is obtained with Post-Token from the concrete sentence included in parentheses.

In Figure 7.4, we give a graphical example of some word representations learned with our model (subsection 7.5.5), obtained either with the Post-Token or the Post-Type. To visualize the representations, we apply

multidimensional scaling.[7] We can see that the model clearly separates between management positions and parts of body, and interestingly, puts "head" closer to management positions, which can be explained by the business and economic nature of the Bllip corpus. The words "chief" and "executive" are located together, yet isolated from others, possibly because of their strong tendency to precede nouns. The arrow on the plot indicates the shift in the meaning when a Post-Token representation is obtained for "head" (part-of-body) within a sentence.

Despite the advantage of Post-Token to account for word senses, we observe that Post-Type performs better in almost all experiments. A likely explanation is that averaging increases the generalizability of representations. For the concrete tasks in which we apply the word representations, the increased robustness simply outweighs context sensitivity. Another explanation could be that Post-Type might be less sensitive to parsing errors at test time.

7.5 Empirical study

7.5.1 Parameters and setup

We train our models following an online learning regime. Specifically, we use the mini-batch step-wise EM (Liang and Klein, 2009; Cappé and Moulines, 2009), and determine the hyper-parameters on the held-out dataset of 10,000 sentences to maximize the log-likelihood. We find out that higher values for the step-wise reduction power α and the mini-batch size lead to better overall log-likelihood, but with a somewhat negative effect on the convergence speed. We have experimented with various values for α ($\{0.6, 0.7, 0.8, 0.9, 1\}$) and the mini-batch size ($\{1, 10, 100, 1000\}$), and finally settle on $\alpha = 1$ and the 1000 sentences in a mini-batch. We find that two iterations over the entire dataset is sufficient to obtain good parameters, cf. Klein (2005).

[7] `https://github.com/scikit-learn/scikit-learn`

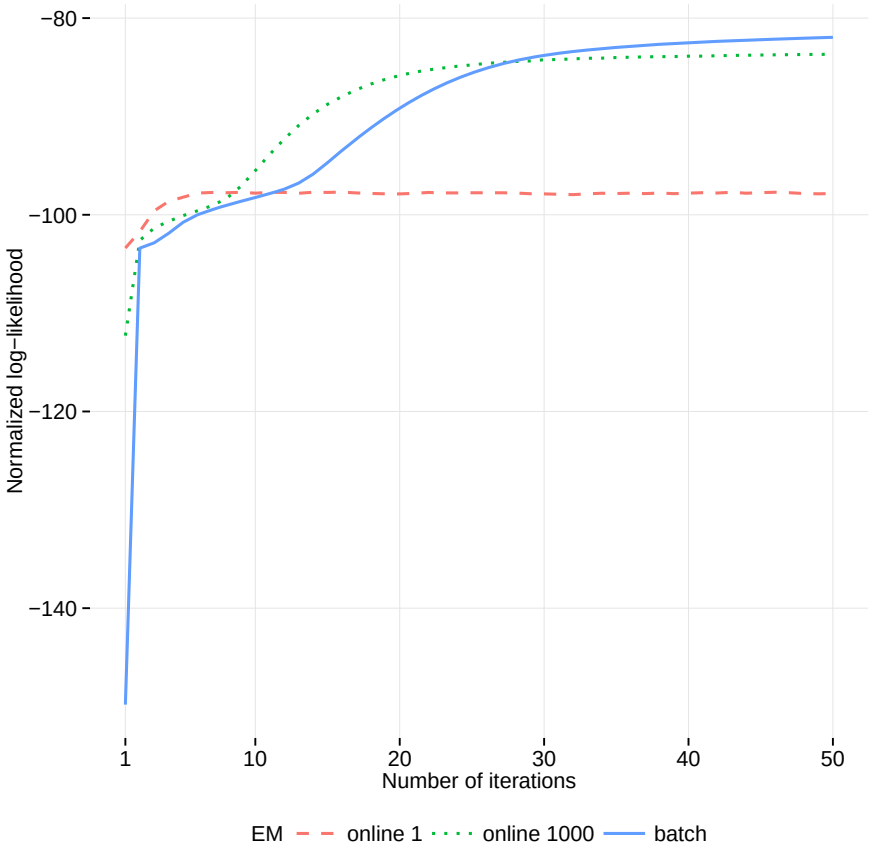


Figure 7.5: Convergence of the batch and the online EM (with a minibatch of either 1 or 1000 sentences, and $\alpha = 1$) for a sequential HMM. The models have been trained on a 10,000-sentence portion of the Dutch training corpus, with the number of states set to 100.

It has been noted in Liang and Klein (2009) that online EM is able to converge to solutions with higher accuracy and log-likelihood on tasks like POS tagging and document classification. To investigate this, we have run a set of experiments. In Figure 7.5, we compare batch and online regimes in

terms of the obtained log-likelihood per iteration. As expected, online EM converges faster, but unlike in the work of Liang and Klein (2009), the solutions in our case perform in fact slightly worse than those found by batch EM beginning at around 30 iterations. We should note, however, that our results are based on a fraction of the training data only, and that a different picture might emerge when taking into account the complete data. Since it is impractical to train a model for 30 epochs or more, we use an online regime with a higher value of α, for which the updates of the old parameters are more conservative. We also find it beneficial to run a handful of iterations of online EM followed by additional two iterations of batch EM (Figure 7.6). The positive effect can be seen for both the sequential and the tree model. They both have similar patterns of convergence, but, although the tree model starts off at a lower log-likelihood, it ultimately succeeds in finding a tighter fit to the data.

Initialization. Since the EM algorithm in our setting only finds a local optimum of the log-likelihood, the initialization of model parameters can have a major impact on the final outcome. We initialize the emission matrices with Brown clusters by first assigning random values between 0 and 1 to the matrix elements, and then multiplying the elements indexed by the words in a cluster by a factor of $f \in \{10, 100, \underline{1K}, 10K\}$. Finally, we normalize the matrices. This technique incorporates a strong bias towards those word-class emissions that exist in Brown clusters. The transition parameters are simply set to random numbers sampled from the uniform distribution between 0 and 1, and finally normalized.

Approximate inference. Following Grave et al. (2013) and Pal et al. (2006), we approximate the belief vectors during inference,[8] which speeds up learning and works as regularization. We use the k-best projection method, in which only k-largest coefficients (in our case $k = \frac{1}{8}N$) are kept.

[8] In a bottom-up pass, a belief vector represents the local evidence by multiplying the messages received from the children of a node, as well as the emission probability at that node.

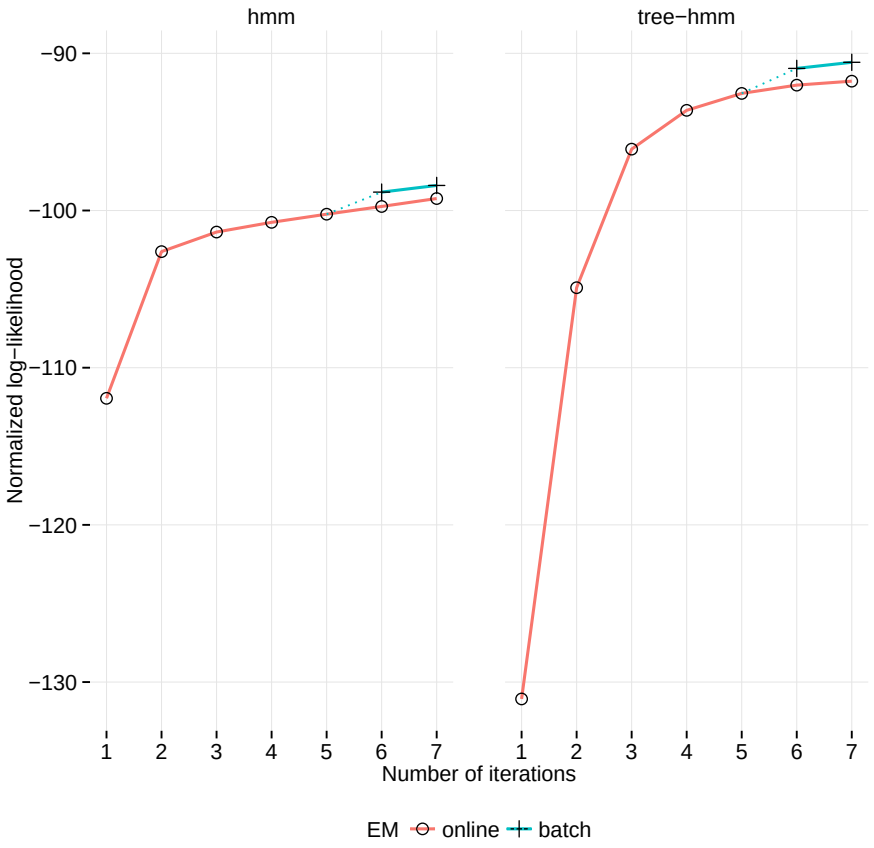


Figure 7.6: Convergence in learning of sequential (*hmm*) and unlabeled tree (*tree-hmm*) HMMs with online EM (mini-batch size set to 1000 sentences). Also shown are the log-likelihoods of batch EM, for which each model is initialized with the parameters from the 5th online iteration, denoted on the plot as dotted lines. The models are trained on a 10,000-sentence portion of the Dutch training corpus, with the number of states set to 80.

7.5.2 Data for obtaining word representations

English. We use the 43M-word Bllip corpus (Charniak et al., 2000) of WSJ texts, from which we remove the sentences in the PTB and those whose length is ≤ 4 or ≥ 40. We use the same parser as in the previous chapter, the MST dependency parser (McDonald and Pereira, 2006) and build a projective, second order model, trained on sections 2–21 of the Penn Treebank WSJ (PTB). The preprocessing steps are the same as those described previously. After parsing, we replace the words occurring less than 40 times with a special symbol to model OOV words. This results in the vocabulary size of around 27,000 words.

Dutch. We first produce a random sample of 2.5M sentences from the SoNaR corpus[9] (Oostdijk et al., 2008), then follow the same preprocessing steps as for English. We parse the corpus with Alpino (van Noord, 2006). In contrast to the English experiments, in which we use word forms, we keep here the root forms produced by the parser's lexical analyzer. The resulting vocabulary size is about 25,000 words. The dependency structures produced by the parser include multiple parents to facilitate the treatment of wh-clauses, coordination and passivization. Since our method expects proper trees, we convert the Alpino output to CoNLL format using `http://www.let.rug.nl/bplank/alpino2conll/`, which also transforms nodes with multiple parents to single-parent nodes.

7.5.3 Evaluation tasks

We evaluate the learned representations in two extrinsic sequence and structure labeling tasks.

7.5.3.1 Named entity recognition

We use the standard CoNLL-2002 shared task dataset for Dutch and CoNLL-2003 dataset for English. We also include the out-of-domain MUC-

[9] `http://lands.let.ru.nl/projects/SoNaR`

7 testset, preprocessed according to Turian et al. (2010).[10] We refer the reader to section 4.2.1 of this thesis and to Ratinov and Roth (2009) for a detailed description of the NER classification problem.

Just like Turian et al. (2010), we use the averaged structured perceptron (Collins, 2002) with Viterbi as the base for our NER system.[11] The classifier is trained for a fixed number of iterations, and uses these baseline features:

- w_k information: is-alphanumeric (= the word includes alphanumeric characters), all-digits (= consists of digits only), all-capitalized (= all letters in the word are capitalized),
  is-capitalized (= only the first letter is capitalized), is-hyphenated (= the word includes a hyphen);
- prefixes and suffixes of w_k of maximum length 3;
- word window: $w_k, w_{k\pm1}, w_{k\pm2}$;
- capitalization pattern in the word window, based on is-capitalized.

We construct N real-valued features for a word vector of dimensionality N, and a simple indicator feature for a categorical word representation.

7.5.3.2 Semantic frame identification

We introduced the task of semantic frame identification (SFI) in Chapter 4. Here, we use the Semafor parser (Das et al., 2014) consisting of two log-linear components trained with gradient-based techniques. The parser is trained and tested on the FrameNet 1.5 full-text annotations. Our test set consists of the same 23 documents as in Hermann et al. (2014). We investigate the effect of word representation features on the frame identification component. We measure Semafor's performance on gold-standard targets, and report the accuracy on *exact matches*, as well as on *partial matches*. The

[10] The number of sentences in the training and test sets is 15,806 (train)/ 5,195 (test) for Dutch, 14,041 (train)/ 3,454 (test) for English and 2417 (test) for MUC-7.

[11] `http://github.com/LxMLS/lxmls-toolkit`

latter give partial credit to identified related frames. We use and modify the publicly available implementation at `http://github.com/sammthomson/semafor`.

Our baseline features for a target w_k include:

- w_k and its parent $w_{\pi(k)}$; if the parent is a preposition, the grandparent is taken by collapsing the dependency),
- the lemmas and POS tags of w_k and $w_{\pi(k)}$,
- syntactic functions of:
  - each of w_k's children,
  - ${w_k}^{12}$,
  - $w_{\pi(k)}$ (towards its parent $w_{\pi(\pi(k))}$.

7.5.4 Baseline word representations

We test our model, which we call SYNFUNC-HMM, against the following baselines:

- BASELINE: use only baseline features and no word representation features
- HMM: use baseline features and word representation features from the sequential HMM
- TREE-HMM: use baseline features and word representation features from the unlabeled-tree HMM

We induce other representations for comparison and include them in addition to the baseline set of features:

- BROWN: use Brown cluster identifiers

[12] This feature was not present in Semafor at the time of our experimental study, yet it improves the test set accuracy by around 0.15.

- DEP-BROWN: use dependency Brown cluster identifiers
- SKIP-GRAM: use Skip-Gram word embeddings

7.5.5 Preparing word representations

Brown clusters. Brown clusters (Brown et al., 1992) are known to be effective and robust when compared, for example, to word embeddings (Bansal et al., 2014; Passos et al., 2014; Nepal and Yates, 2014; Qu et al., 2015). The method can be seen as a special case of a HMM in which word emissions are deterministic, i.e. a word belongs to at most one semantic class. While the standard algorithm relies on sequential, bigram data representation, we also use dependency-based clusters following our proposed extension from Chapter 6 (Šuster and van Noord, 2014). For both methods, we use the publicly available implementations.[13]

Following other work on English (Koo et al., 2008; Nepal and Yates, 2014), we add both coarse- and fine-grained clusters as features by using prefixes of length 4, 6, 10 and 20 in addition to the complete binary tree path. For Dutch, coarser-grained clusters do not yield any improvement. Brown features are included in a window around the target word, just as the NER word features. When adding cluster features to the frame-semantic parser, we transform the cluster identifiers to one-hot vectors, which gives a small improvement over the use of indicator features.

HMM-based models. The N-dimensional representations obtained from HMMs and their variants are included as N distinct continuous features. In the NER task, word representations are included at w_k and w_{k+1} for Dutch and at w_k for English, which we determined on the NER development sets. We investigate state space sizes of 64, 128 and 256 and finally choose $N{=}128$ as a reasonable trade-off between training time and quality. We use the same dimensionality for other word representation models in this chapter.

[13] `http://github.com/percyliang/brown-cluster`, `http://github.com/rug-compling/dep-brown-cluster`

We observe that by constraining SYNFUNC-HMM to use only the k most frequent syntactic functions and to treat the remaining ones as a single special syntactic function, we obtain better results on the development part of the NER datasets. This is likely because for a model with all S syntactic functions produced by the parser, there is less learning evidence for more infrequent syntactic functions. We explore the effect of keeping up to five most frequent syntactic functions, ignoring functional ones such as punctuation and determiner.[14] The final selection is shown in Table 7.1.

English	Dutch
nmod (nominal modifier)	mod (modifier)
pmod (prepositional modifier)	su (subject)
sub (subject)	obj1 (direct object)
	cnj (conjunction)
	mwp (multiword unit)

Table 7.1: Syntactic functions in SYNFUNC-HMM for English (produced by the MST parser) and Dutch (produced by Alpino).

For NER experiments, the representations from each HMM model are obtained with three different "decoding" methods described in subsection 7.4.2. In the evaluation, we only report the results for the method that performed best on the NER development sets, POST-TYPE.[15]

Word embeddings. We use the Skip-Gram model presented in Mikolov et al. (2013a) (`https://code.google.com/p/word2vec/`), trained with negative sampling (Mikolov et al., 2013b). The training seeks to maximize the dot product between word-context pairs encountered in the training corpus and minimize the dot product between those pairs in which the context word is randomly sampled. We set both the number of negative examples

[14] We define the list of function-marker syntactic functions following Goldberg and Orwant (2013).

[15] While exploring the constraint on the number of syntactic functions, we do find that POST-TOKEN outperforms POST-TYPE in some sets of syntactic functions, but not in the final selection.

and the size of the context window to 5, the down-sampling threshold to 1×10^{-4}, and the number of iterations to 15.

7.5.6 NER results

The results for English and Dutch CoNLL test sets are shown in Table 7.2. For **English**, all HMM-based models improve the baseline, with the sequential HMM achieving the highest F-score. Our SYNFUNC-HMM performs on a par with SKIP-GRAM. It outperforms the unlabeled-tree model, which is an indication that the added observations are meaningful. Brown clusters do not exceed the BASELINE score.[16] Testing for significance with a bootstrap method (Søgaard et al., 2014), we find out that only HMM significantly (at $p < 0.01$) improves over BASELINE on macro-F1, while SKIPGRAM and SYNFUNC-HMM show significant improvements only for the location entity type.

	English			Dutch		
Model	P	R	F-1	P	R	F-1
BASELINE	80.12	77.30	78.69	75.36	70.92	73.07
HMM	**81.49**	78.90	**80.17**	77.61	73.97	75.74
TREE-HMM	80.49	78.10	79.28	77.41	73.48	75.40
SYNFUNC-HMM	80.65	78.90	79.76 (+.48)	**78.54**	**75.23**	**76.85** (+1.45)
BROWN	80.15	77.28	78.70	77.88	71.73	74.68
DEP-BROWN	78.80	75.73	77.23	77.50	73.66	75.53
SKIP-GRAM	80.80	**78.98**	79.88	76.02	71.28	73.57

Table 7.2: NER results (precision, recall and F-score) on English and Dutch test sets. Best result per column in bold. The score increase reported in parentheses is in comparison to TREE-HMM.

The general trend for **Dutch** is somewhat different. Most notably, all word representations contribute much more effectively to the overall classification performance compared to English. The best-scoring model, our

[16] However, after additional experiments we observe that the cluster features do improve over the baseline score when the number of clusters is increased.

Synfunc-Hmm, improves over the baseline significantly, by as much as 3.8 points. Part of the reason Synfunc-Hmm works so well in this case is that it can make use of the informative "mwp" syntactic function between the parts of a multiword unit. Similarly as for English, the unlabeled-tree model performs slightly worse than the sequential Hmm. The cluster features are more valuable here than in English, and we also observe a 0.7-point advantage by using dependency Brown clusters over the standard, bigram Brown clusters. The Skip-Gram model does not perform as well as in English, which might indicate that the hyper-parameters would need fine-tuning specific to Dutch.

On the **out-of-domain** MUC dataset (Figure 7.3), the tree-based representations appear to perform poorly, whereas the highest score is achieved by the Skip-Gram method. Unfortunately, it is difficult to generalize from the F-1 result alone. Concretely, the dataset contains 3,518 named entities, and the Skip-Gram method makes 258 correct predictions more than Tree-Hmm. However, because the MUC dataset covers the narrow topic of missile-launch scenarios, the system gets badly penalized if a mistake is made repeatedly for a certain named entity. For example, only the entity "NASA" occurs 103 times, mostly wrongly classified by the Tree-Hmm system, but correctly by Skip-Gram. The overall performance may therefore hinge on a limited number of frequently occurring entities. A workaround is to evaluate per *entity type*, i.e. calculate the F-score for each entity, then average over all entity types. The results for this evaluation scenario are reported as F-1_{type}. Skip-Gram still performs best, but the difference to other models is smaller. Finally, we also report a third F-1 score, $\text{F-1}_{\text{unlab}}$, which is calculated just like F-1_{type} but ignoring the actual entity label. So, if a named-entity token is recognized as such, we count it as correct prediction ignoring the entity label type, similarly as done by Ratinov and Roth (2009). Since this setting boosts the performance of Synfunc-Hmm, we can conclude that the features obtained from Synfunc-Hmm are more effective at identifying entities rather than at labeling them.

The fact that we observe different tendencies for English and Dutch

	MUC test set		
Model	F-1	F-1_{type}	$\text{F-1}_{\text{unlab}}$
Baseline	65.44	87.04	96.25
Hmm	70.20	87.66	96.50
Tree-Hmm	65.67	86.99	96.53
Synfunc-Hmm	66.49 (+.82)	86.93 (-.06)	**96.69** (+.16)
Brown	68.85	87.72	96.67
Dep-Brown	68.31	87.44	96.47
Skip-Gram	**72.42**	**88.94**	**96.69**

Table 7.3: NER results on the English MUC dataset. Best result per column in bold. The score increase reported in parentheses is in comparison to Tree-Hmm. F-1_{type} is the F-score measured per word type, and $\text{F-1}_{\text{unlab}}$ is the F-score measured per word type, ignoring labels.

can be attributed to an interplay of factors, such as language differences (Bender, 2011), difference in accuracy of syntactic parsers, and differences specific to the evaluation datasets. We briefly discuss the first possibility. It is clear from Table 7.2 that all syntax-based models (Dep-Brown, Tree-Hmm, Synfunc-Hmm) generally benefit Dutch more than English. Along similar lines as in section 6.6.2, we hypothesize that since the word order in Dutch is generally less fixed than in English,[17] a sequence-based model for Dutch cannot capture selectional preferences that successfully, i.e. there is more interchanging of semantically diverse words in a small word window. This then may make the difference in performance between sequential and tree models more apparent for Dutch.

7.5.7 SFI results

We now turn to the results for semantic frame identification. Here, the results are for English only due to the lack of such frame-annotated resources for Dutch. The results are shown in Table 7.4. None of the models

[17] For instance, it is unusual for the direct object in English to precede the verb, but quite common in Dutch.

clearly outperforms other models. The highest score is obtained with the Skip-Gram embeddings, however, the difference to other models is small. Concretely, Skip-Gram correctly identifies only two cases more than Dep-Brown, out of 3681 correctly disambiguated frames. The Synfunc-Hmm model yields a noticeable improvement over the Hmm and Baseline models with both scorings, and over the Tree-Hmm with partial scoring.

Finally, some notes on statistical significance of these results. We have tested for significance by running a paired permutation test. On exact matches, only Dep-Brown and Brown significantly outperform the baseline with the $p < 0.05$. On partial matches, Dep-Brown, Brown, Skip-Gram and Synfunc-Hmm all outperform the baseline significantly. Synfunc-Hmm performs significantly better than Tree-Hmm on partial matches, whereas the difference between Skip-Gram and Synfunc-Hmm is not significant.

Model	Exact	Partial
Baseline	82.70	90.44
Hmm	82.20	90.20
Tree-Hmm	82.89	90.59
Synfunc-Hmm	82.95 (+0.06)	90.80 (+0.21)
Brown	83.15	90.74
Dep-Brown	83.15	90.76
Skip-Gram	**83.19**	**90.91**

Table 7.4: Frame identification accuracy. Score increase in parentheses is relative to Tree-Hmm.

7.5.8 Further discussion

To summarize our findings about the effect of using word representations obtained from labeled or unlabeled tree models, we have seen that in the NER experiments *unlabeled* syntactic trees do not in general provide a better structure for defining the contexts compared to plain sequences. The only exception is the case of dependency Brown clustering for Dutch. Comparing our results to those of Grave et al. (2013), we therefore cannot confirm

the same advantage when using unlabeled-tree representations. We reflect on a possible reason in the next section. In SFI, however, the unlabeled-tree representations do compare more favorably to sequential representations.

How important is the addition of syntactic functions in the tree HMM model? In practically all experiments, the extension with syntactic functions has an advantage over the baseline and other HMM-based representations. In Dutch NER, this advantage extends also over all other models we compare against. The explanation for these improvements appears to lie in discrimination between the types of contexts, for example between a modifier and a subject, which is impossible in sequential or unlabeled-tree HMM architectures.

As methods for comparison, we have also included Brown and dependency Brown clusters as baseline features for the downstream tasks. In this way, we have provided additional evidence that confirms our findings from the previous chapter, namely that dependency Brown clustering applied to Dutch is superior to standard Brown clustering, yet when applied to English we observe an opposite effect. We have argued that the reason might lie in language differences (such as word order), differences in accuracy of the parsers and in the specifics of the evaluation datasets.

7.6 Additional related work

HMMs have been used successfully for learning word representations already before, see Huang et al. (2014) for an overview, with an emphasis on investigating domain adaptability. Among the models they study, three techniques provide context-sensitive representations, meaning that the representation and the features obtained from it depend on the local context around the target word. Their first model is a Naive Bayes model with categorical hidden states, trained with EM and based on trigrams, without taking into account the dependencies between the trigrams; the second model is a HMM trained with EM, using the Max-Product and Post-Token methods (see section 7.4.2) for constructing the representations.

This model correspond to our sequential Hmm, except that our training procedure uses several refinements (see section 7.4.1 and section 7.5.1). Their third model is a latent-variable model called Partial Lattice Markov Random Field, consisting of several layers of hidden states. The inference in this model is made tractable by leaving out some of the connections between the hidden states. One of their findings is that the second and the third model do particularly well on polysemous words when evaluated in chunking and POS tagging.

A model architecture similar to Huang et al. (2014)'s third model is explored by Nepal and Yates (2014), and belongs to the family of factorial HMMs. The emphasis in their work is on developing a practical learning mechanism via variational approximation. They show that their model outperforms simpler models like HMMs and Brown clusters on chunking and tagging tasks, possibly due to the property of the factorial HMM that it is sensitive to the entire word sequence. The downside of the factorial architecture, however, is that it does not scale well to large datasets. For example, Nepal and Yates need to restrict the training dataset size to around 110,000 sentences to make the training practical.

The extension of HMMs to dependency trees for the purpose of word representation learning was first proposed by Grave et al. (2013). Although our baseline HMM methods, Hmm and Tree-Hmm, conceptually follow the models of Grave et al., there are still several differences.[18] One source of differences is in the precise steps taken to refine the learning procedure, such as when performing Brown initialization, state splitting, and also approximation of belief vectors during inference. Another source involves the evaluation setting. Their NER classifier uses only a single baseline feature, and when including Brown clusters, they do not make use of the clustering hierarchy to produce coarser-grained cluster features. In general, we can say that Brown clusters provide for more competitive features in our

[18]Their implementation is not publicly available, so we were unable to replicate their results using their source code. Our comparison to their method is based on the published work and personal communication with the authors.

work than in the work of Grave et al., which is probably connected to our use of differently grained clusters. In this respect, our experimental setting is more similar to Turian et al. (2010). Another practical difference is that Grave et al. concatenate words with POS tags to construct the input text, whereas we use tokens (English) or word roots (Dutch). Although the concatenation with POS tags is a preprocessing step, it introduces in our opinion a strong bias that allow easier discrimination between the cases of homonymy.

There is a long tradition of unsupervised training of HMMs for POS tagging (Kupiec, 1992; Merialdo, 1994), with more recent work on incorporating bias by favoring sparse posterior distributions within the posterior regularization framework (Graça et al., 2007), and for example on auto-supervised refinement of HMMs (Garrette and Baldridge, 2012). It would be interesting to see how well these techniques apply to word representation learning methods like ours. Another interesting area for exploration is the compositionality of latent-variable representations. The recent work by Grave et al. (2014) propose compositionality techniques for HMM-based models that originate in distributional-semantics research.

The incorporation of word representations into semantic frame identification has been explored in Hermann et al. (2014). They perform a projection of generic word embeddings for context words to a low-dimensional representation, which also learns an embedding for each frame label. The method selects the frame closest to the low-dimensional representation obtained through mapping of the input embeddings. Their approach differs from ours in that they induce new representations that are tied to a specific application, whereas we aim to obtain linguistically enhanced word representations that can be subsequently used in a variety of tasks. In our case, the word representations are thus included as additional features in the log-linear model. Although Hermann et al. also use syntactic functions, they are used to position the general word embeddings within a single input context embedding, whereas we use the syntactic functions explicitly during representation learning. Unfortunately, we are unable to directly

compare our results with theirs as their parser implementation is proprietary. The accuracy of our baseline system on the test set is 0.27 percent lower in the exact matching regime and 0.07 lower in the partial matching regime compared to the baseline implementation score reported in Hermann et al. (2014) using the implementation of Das et al. (2014).[19]

The topic of context type (syntactic vs. linear) has been abundantly treated in the literature on distributional semantic models (Lin, 1998a; Baroni and Lenci, 2010; van de Cruys, 2010) and elsewhere (Boyd-Graber and Blei, 2008; Tjong Kim Sang and Hofmann, 2009; Séaghdha and Korhonen, 2014). In the framework of word embeddings, Levy and Goldberg (2014a) generalize the Skip-Gram model to arbitrary contexts, and study dependency-based contexts on a set of intrinsic tasks. Wang et al. (2015) build on the work of Levy and Goldberg (2014a) and propose a relation-dependent model for learning word embeddings. The idea that different contexts are differently important during model learning has been explored, for example, by Ling et al. (2015), who use an attention model for weighting the relevant context words.

7.7 Conclusion and future work

In this chapter, we have proposed an extension of a Hidden Markov tree model with syntactic functions. We have seen that by adopting a dependency-language model and keeping the syntactic functions associated to dependency trees, the model can discriminate better between the word contexts that are used as input compared to the sequential or the unlabeled-tree HMMs. In our empirical study, we demonstrate a positive effect of the proposed word representations on the performance of two NLP applications, named entity recognition and semantic frame identification. Since we use the sequential and the unlabeled tree models as base-

[19] Among other implementation differences, they introduce a variable capturing lexical semantic relations from WordNet.

lines, we observe that blindly preferring the former over the latter does not always lead to an improvement.

One promising direction for future work is how to discriminate between context types so that more accurate representation models can be built in other frameworks. Another interesting direction is how to compose the word-level representation into higher-order representations, such as phrases and sentences. Although some work exists for sequential HMMs (Grave et al., 2014), it would be interesting to see how the tree structure can guide semantic composition in labeled- or unlabeled-tree models. In this chapter, as well as in the previous Chapter 6, we have proposed models whose learning input is based on the syntactic structures produced by the parser. Although the automatic syntactic analysis can mostly be very accurate, it sometimes leads to mistakes. Wrongly selected attachment sites and dependency labels possibly have a negative effect on the obtained word representations. Therefore, it would be interesting to investigate this further, preferably by distinguishing between the effect of parser errors in the word representation training corpus and at obtaining the word representations on the parsed test set. If it is true that parsing mistakes are detrimental for the quality of word representations, then adding parse reliability information to the model could be one possible solution: For example—assuming that longer attachments are parsed less reliably by the parser—one could add a dependency length bias in form of context weighting to the model (cf. Smith and Eisner (2006)).

PART IV

Learning word representations in a multilingual context

CHAPTER 8

Bilingual learning of multi-sense embeddings with discrete autoencoders

We present an approach to learning multi-sense word embeddings relying both on monolingual and bilingual information. Our model consists of an encoder, which uses monolingual and bilingual context (i.e. a parallel sentence) to choose a sense for a given word, and a decoder which predicts context words based on the chosen sense. The two components are estimated jointly. We observe that the word representations induced from bilingual data outperform the monolingual counterparts across a range of evaluation tasks, even though crosslingual information is not available at test time.

8.1 Introduction

Approaches to distributional learning of word embeddings (introduced in section 3.3) have received much attention in recent years, and the induced

representations have been shown to capture syntactic and semantic properties of words. They have been evaluated intrinsically (Mikolov et al., 2013a; Baroni et al., 2014; Levy and Goldberg, 2014b) and have also been used in concrete NLP applications to deal with word sparsity and improve generalization (Turian et al., 2010; Collobert et al., 2011; Bansal et al., 2014; Passos et al., 2014). While most work has focused on developing embedding models which represent a word with a single vector, some researchers have attempted to capture *polysemy* explicitly and have encoded properties of each word with multiple vectors (Huang et al., 2012; Tian et al., 2014; Neelakantan et al., 2014; Chen et al., 2014; Li and Jurafsky, 2015).

In parallel to this work on multi-sense word embeddings, another line of research has investigated integrating *multilingual* data, with largely two distinct goals in mind. The first goal has been to obtain representations for several languages in the shared parameter space, which then enables the transfer of a downstream model (e.g., a syntactic parser) trained on annotated training data in one language to another language lacking this annotation. Some representative works are Klementiev et al. (2012), Hermann and Blunsom (2014), Gouws et al. (2014) and Chandar A P et al. (2014). Secondly, information from another language can also be leveraged to yield better first-language embeddings (Guo et al., 2014). Our approach falls in this latter, much less explored category.

More abstractly, the work presented here can be characterized by a view of multilingual learning as a means of language grounding. Although the term "grounding" is normally used only when incorporating other modalities (like vision, for example) in language processing, some researchers nevertheless use it also when working in a multilingual scenario. Of course, multilingual grounding is not constrained to representation learning, but can apply to any ML problem which satisfies the condition that a given language is thought to benefit from other languages (Faruqui and Dyer, 2014b; Zou et al., 2013; Titov and Klementiev, 2012b; Snyder and Barzilay, 2010; Naseem et al., 2009).

The main idea in this chapter rests on the intuition that polysemy in

one language can be at least partially resolved by looking at the translation of the word and its context in another language (Kaji, 2003; Ng et al., 2003; Diab and Resnik, 2002; Ide, 2000; Dagan and Itai, 1994; Brown et al., 1991). Better sense assignments can then lead to better sense-specific word embeddings.

We propose a model that uses second-language embeddings as a supervisory signal in learning multi-sense representations in the first language. This supervision is easy to obtain for many language pairs as numerous parallel corpora exist nowadays. Our model, which can be seen as an autoencoder with a discrete hidden layer encoding word senses, leverages bilingual data in its encoding part, while the decoder predicts the surrounding words relying on the predicted senses. Although we expect a parallel corpus that is word-aligned as input, we also investigate a set-up which is indicative of the model performance when only sentence alignment is available. We explain our approach in detail in the next sections.

8.2 Contributions

In Chapter 1, we stated the research question for this last part of the thesis as:

- How can bilingual, parallel corpora be employed for better multi-sense word representations?

In summary, we can answer the question with the following findings:

- The second-language signal effectively improves the quality of multi-sense embeddings as seen on a variety of intrinsic tasks for English, with the results superior to that of the baseline Skip-Gram model, even though the crosslingual information is not available at test time.

- This finding is robust across several settings, such as varying embedding dimensionality, vocabulary size and amount of data.

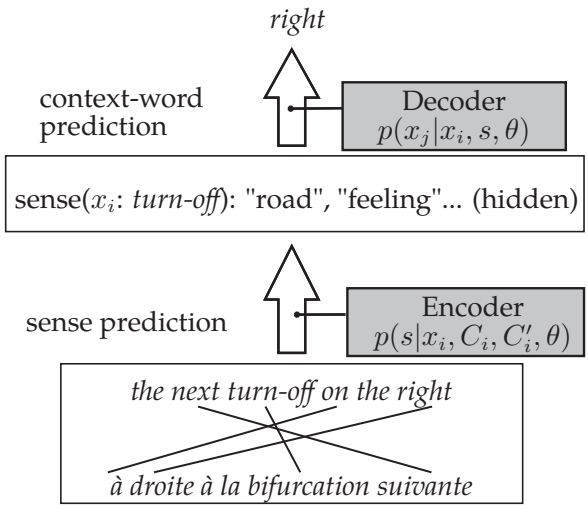


Figure 8.1: Model schema: The sense encoder uses bilingual signal in its predictions, and the decoder predicts context words based on those. Here, sense prediction concerns the word "turn-off"; in the decoding part, surrounding words are predicted, one at a time (e.g. "right"). The learning is performed jointly.

- In the extrinsic POS tagging task, the second-language signal also offers improvements over monolingually-trained multi-sense embeddings, however, the standard Skip-Gram embeddings turn out to be the most robust in this task.

8.3 Word embeddings with discrete autoencoders

Our method borrows its general structure from neural autoencoders (Rumelhart et al., 1986; Bengio et al., 2013). Autoencoders are networks trained to reproduce their input by first mapping their input to a (lower dimensional) hidden layer and then predicting an approximation of the input relying on this hidden layer. In our case, the hidden layer is not

a real-valued vector, but is a categorical variable encoding the sense of a word. Discrete-state autoencoders have been successful in several natural language processing applications, including POS tagging and word alignment (Ammar et al., 2014), semantic role induction (Titov and Khoddam, 2015) and relation discovery (Marcheggiani and Titov, 2016).

More formally, our model consists of two components: an *encoding* part which assigns a sense to a pivot word, and a *reconstruction* (decoding) part recovering context words based on the pivot word and its sense. As predictions are probabilistic ("soft"), the reconstruction step involves summation over all potential word senses. The goal is to find embedding parameters which minimize the error in recovering context words based on the pivot word and the sense assignment. Parameters of both encoding and reconstruction are jointly optimized. Intuitively, a good sense assignment should make the reconstruction step as easy as possible. The encoder uses not only words in the first-language sentence to choose the sense but also, at training time, is conditioning its decisions on the words in the second-language sentence. We hypothesize that the injection of crosslingual information will guide learning towards inducing more informative sense-specific word representations. Consequently, using this information at training time would benefit the model even though crosslingual information is not available to the encoder at test time.

We specify the encoding part as a log-linear model:

$$p(s|x_i, C_i, C'_i, \theta) \propto \exp\big(\varphi_{i,s}^{\top}\big(\frac{1-\lambda}{|C_i|}\sum_{j\in C_i}\gamma_j + \frac{\lambda}{|C'_i|}\sum_{k\in C'_i}\gamma'_k\big)\big). \tag{8.1}$$

To choose the sense $s \in \mathcal{S}$ for a word x_i, we use the bag of context words C_i from the first language l, as well as the bag of context words C'_i from the second language l'.[1] The first-language context C_i is defined as a multiset $C_i = \{x_{i-n}, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{i+n}\}$ including words around the pivot word in the window of size n to each side. We set n to 5 in all our ex-

[1]We have also considered a formulation which included a sense-specific bias $b_{x_i,s} \in \mathbb{R}$ to capture relative frequency of latent senses but it did not seem to affect performance.

periments. The crosslingual context C'_i is discussed in section 8.4, where we rely on word alignments to choose context of different sizes. We distinguish between sense-specific embeddings, denoted by $\varphi \in \mathbb{R}^d$, and generic sense-agnostic ones, denoted $\{\gamma, \gamma'\} \in \mathbb{R}^d$ for first and second language, respectively. The number of sense-specific embeddings is the same for all words. We use θ to denote all these embedding parameters. They are learned jointly from scratch, with the exception of the second-language embeddings, which we first pre-train on the given language.

The hyperparameter $\lambda \in \mathbb{R}, 0 \leq \lambda \leq 1$ weights the contribution of the first and the second language. Setting $\lambda = 0$ would drop the second-language component and use only the first language. Our formulation allows the addition of new languages easily, provided that the second-language embeddings live in the shared semantic (i.e. embedding) space.

The role of the reconstruction part is to predict a context word x_j given the pivot x_i and the current estimate of its s:

$$p(x_j|x_i, s, \theta) = \frac{\exp(\varphi_{i,s}^\top \gamma_j)}{\sum_{k \in |\mathcal{V}|} \exp(\varphi_{i,s}^\top \gamma_k)}, \tag{8.2}$$

where $|\mathcal{V}|$ is the vocabulary size. This is effectively a Skip-Gram model (Mikolov et al., 2013a) extended to rely on senses.

8.3.1 Learning and regularization

As sense assignments are not observed during training, the learning objective includes marginalization over word senses and thus can be written as:

$$\sum_i \sum_{j \in C_{x_i}} \log \sum_{s \in \mathcal{S}} p(x_j|x_i, s, \theta) p(s|x_i, C_i, C'_i, \theta), \tag{8.3}$$

in which index i goes over all pivot words in the first language, j over all context words to predict at each i, and s marginalizes over all possible senses of the word x_i. In practice, we avoid the costly computation of the normalization factor in the softmax computation of equation (8.2) and

use negative sampling[2] (Mikolov et al., 2013b) instead of $\log p(x_j|x_i, s, \theta)$:

$$\log \sigma(\varphi_{i,s}^\top \gamma_j) + \sum_{x \in N} \log \sigma(-\varphi_{i,s}^\top \gamma_x), \tag{8.4}$$

where σ is the sigmoid non-linearity function $1/1+e^{-z}$, and γ_x is a word embedding from the sample of negative (noisy) words N. Optimizing the autoencoding objective is broadly similar to the learning algorithm defined for multi-sense embedding induction in some of the previous work (Neelakantan et al., 2014; Li and Jurafsky, 2015). Note though that this previous work has considered only monolingual context.

We use a minibatch training regime and seek to optimize the objective function $L(\mathcal{B}, \theta)$ for each minibatch $\mathcal{B}$. We found that optimizing this objective directly often resulted in inducing very flat posterior distributions. We therefore use a form of posterior regularization (Ganchev et al., 2010) where we can encode our prior expectations that the posteriors should be sharp. The regularized objective for a minibatch is defined as

$$L(\mathcal{B}, \theta) + \lambda_H \sum_{i \in \mathcal{B}} H(q_i), \tag{8.5}$$

where H is the entropy function and q_i are the posterior distributions from the encoder ($p(s|x_i, C_i, C_i', \theta)$). This modified objective can also be motivated from a variational approximation perspective, but we refer the reader to Marcheggiani and Titov (2016) for details. By varying the parameter $\lambda_H \in \mathbb{R}$, it is easy to control the amount of entropy regularization. For $\lambda_H > 0$, the objective is optimized with flatter posteriors (the sum of entropies must be as large as possible), while $\lambda_H < 0$ infers more peaky posteriors. When $\lambda_H \to -\infty$, the probability mass needs to be concentrated

[2]The idea behind negative sampling is simple: We would like to contrast the correct (positive) instances—in our case the pivot-context pair actually observed—with a pair in which the context word is some random, negative word that (very likely) does not fit the context well. In other words, we are contrasting plausible word combinations with implausible ones (Smith and Eisner, 2005).

on a single sense, resulting in an algorithm similar to hard EM. In practice, we found that using hard-update training[3], which is closely related to the $\lambda_H \to -\infty$ setting, led to best performance.

8.3.2 Obtaining word representations

At test time, we construct the word representations by averaging all sense embeddings for a word x_i and weighting them with the sense expectations[4] (Li and Jurafsky, 2015):

$$\omega_i = \sum_{s \in \mathcal{S}} p(s|x_i, C_i)\varphi_{i,s}. \tag{8.6}$$

Unlike in training, the sense prediction step here does not use the crosslingual context C'_i since it is not available in the evaluation tasks. In our treatment of the problem, instead of marginalizing out the unobservable crosslingual context, we simply ignore it in computation.

Sometimes, even the first-language context is missing, as is the situation in many word similarity tasks. In that case, we just use the uniform average, $1/|\mathcal{S}| \sum_{s \in \mathcal{S}} \varphi_{i,s}$.

8.4 Word affiliation from alignments

In defining the crosslingual signal we draw on a heuristic inspired by Devlin et al. (2014). The second-language context words are taken to be the multiset of words around and including the pivot affiliated to x_i:

$$C'_i = \{x'_{a_i-m}, ..., x'_{a_i}, ..., x'_{a_i+m}\}, \tag{8.7}$$

where x'_{a_i} is the word affiliated to x_i and the parameter m regulates the context window size. By choosing $m = 0$, only the affiliated word is used

[3] I.e. updating only that embedding φ_{i,s^*} for which $s^* = \arg\max_s p(s|x_i, C_i, C'_i, \theta)$.

[4] Although our training objective has sparsity-inducing properties, the posteriors at test time are not entirely peaked, which makes weighting beneficial.

as l' context, and by choosing $m = \infty$, the l' context is the entire sentence (≈uniform alignment). To obtain the index a_i, we use the following rules:

1) If x_i aligns to exactly one second-language word, a_i is the index of the word it aligns to.
2) If x_i aligns to multiple words, a_i is the index of the aligned word in the middle (and rounding down when necessary).
3) If x_i is unaligned, C'_i is empty, therefore no l' context is used.

We use the cdec aligner (Dyer et al., 2010) to word-align the parallel corpora.

8.5 Parameters and set-up

8.5.1 Learning parameters

We use AdaGrad as the gradient-based optimization algorithm (Duchi et al., 2011) with initial learning rate set to 0.1. This techniques adapts the learning rate in such a way that larger updates are performed for infrequent parameters and smaller updates for frequent parameters. AdaGrad is thought to be well-suited for situations with sparse data, which is exactly the kind found in language processing. Intuitively, we would like to use larger updates for rare words (i.e. their embeddings) and smaller updates for frequent words.

We set the minibatch size to 1000, the number of negative samples to 1, the factor for sub-sampling of frequent words to 0.001 and the window size parameter m to 5. All the embeddings are 50-dimensional unless specified otherwise. They are initialized by sampling from the uniform distribution between $[-0.05, 0.05]$. We include in the vocabulary all words occurring in the corpus at least 20 times unless specified differently. We set the number of senses per word to 3, and discuss the importance of this choice in subsection 8.7.4 and section 8.9. All other parameters with their default values can be examined in the source code available online.

8.5.2 Bilingual data

In a large body of work on multilingual word representations, Europarl (Koehn, 2005) is the preferred source of parallel data. However, the domain of Europarl is rather constrained, whereas we would like to obtain word representations of more general language, also to carry out an effective evaluation on semantic similarity datasets where domains are usually broader. We therefore use the following parallel corpora: News Commentary (Bojar et al., 2013) (abbreviated NC), Yandex-1M[5] (RU-EN), CzEng 1.0 (Bojar et al., 2012) (CZ-EN), from which we exclude the EU legislation texts, and GigaFrEn (Callison-Burch et al., 2009) (FR-EN). The sizes of the corpora are reported in Table 8.1. The word representations trained on the NC corpora are evaluated only intrinsically due to the small sizes.

Corpus	Language	Words	Sent.
NC	Fr	4 M	0.2 M
NC	Ru	4 M	0.2 M
NC	Cz	3 M	0.1 M
NC	De	4 M	0.2 M
NC	Es	4 M	0.2 M
RU-EN	Ru	24 M	1 M
CZ-EN	Cz	126 M	10 M
FR-EN	Fr	670 M	23 M

Table 8.1: Parallel corpora used in this study. The word sizes reported are based on the English part of the corpus. Each language pair in NC has a different English part, hence the varying number of sentences per target language.

[5] https://translate.yandex.ru/corpus

8.6 Evaluation tasks

We evaluate the quality of our word representations on a number of tasks, both intrinsic and extrinsic.[6] We describe those next.

8.6.1 Word similarity

We are interested here in how well the semantic similarity ratings obtained from embedding comparisons correlate to human ratings, which is an important indicator of the generic quality of word embeddings (Schnabel et al., 2015). For this purpose, we use a variety of similarity benchmarks for English and report the Spearman ρ correlation scores between the human ratings and the cosine ratings obtained from our word representations. The SCWS benchmark (Huang et al., 2012) is probably the most suitable similarity dataset for evaluating multi-sense embeddings, since it allows us to perform the sense prediction step based on the sentential context provided for each word in the pair, as described in section 4.1.

We also use several other benchmarks, but they only provide the ratings for the word pairs without context. WS-353 contains 353 human-rated word pairs (Finkelstein et al., 2001), while Agirre et al. (2009) separate this benchmark for similarity (WS-SIM) and relatedness (WS-REL). The RG-65 (Rubenstein and Goodenough, 1965) and the MC-30 (Miller and Charles, 1991) benchmarks contain nouns only. The MTurk-287 (Radinsky et al., 2011) and MTurk-771 (Halawi et al., 2012) include word pairs whose similarity was crowdsourced from AMT. Similarly, MEN (Bruni et al., 2012) is an AMT-annotated dataset of 3000 word pairs. The YP-130 (Yang and Powers, 2006) and Verb-143 (Baker et al., 2014) measure verb similarity. Rare-Word (Luong et al., 2013) contains 2034 rare-word pairs. Finally, SimLex-999 (Hill et al., 2014b) is intended to measure pure similarity as opposed to relatedness. For these benchmarks, we prepare the word representations by taking a uniform average of all sense embeddings per word. The eval-

[6] The evaluation code and the implementation of the models can be found online, see Chapter 1.

uation is carried out using the tool described in Faruqui and Dyer (2014a). Out of readability concerns, we report the results by averaging over all benchmarks (Similarity), and include the individual results in Appendix C.

8.6.2 Supersense similarity

We also evaluate on a task measuring the similarity between our embeddings (uniformly averaged in the case of multi-sense embeddings) and a matrix of supersense features extracted from the English SemCor. We perform the evaluation using the Qvec tool (Tsvetkov et al., 2015). We choose this method because it has been shown to output scores that correlate well with extrinsic tasks, e.g. text classification and sentiment analysis. We believe that this, in combination with word similarity tasks from the previous section, can give a reliable picture of the generic quality of word embeddings studied in this chapter.

8.6.3 POS tagging

As our downstream evaluation task, we use the learned word representations at the embedding layer of a neural network tagging model. We use the same convolutional architecture as Li and Jurafsky (2015): an input layer taking a concatenation of neighboring embeddings as input, three hidden layers with a rectified linear unit activation function and a softmax output layer. No other features are used by the tagging model. We train for 10 epochs using one sentence as a batch. Other more specialized hyperparameters can be examined in the source code. The multi-sense word embeddings are inferred from the sentential context using the weighted average, just as in the evaluation on the SCWS dataset. We use the standard splits of the Wall Street Journal portion of the Penn Treebank: 0–18 for training, 19–21 for development and 22–24 for testing.

8.7 Results

8.7.1 Overview

We compare three embeddings models: Skip-Gram (SG), Multi-sense (MU) and Bilingual Multi-sense (BIMU). We use our own implementation for each of them. The first two can be seen as simpler variants of the BIMU model: In SG we omit the encoder entirely, and in MU we omit only the second-language (l') part of the encoder in equation (8.1). We train the SG and the MU models on the English part of the parallel corpora. Separate SG models are trained on each of the second-language parts to obtained pre-trained embeddings which are then used by the BIMU model. During experiments, we keep those parameters that are common to all methods fixed. The values λ and m for controlling the second-language signal in BIMU are set on the POS tagging development set (cf. subsection 8.8.1).

The results on the SCWS benchmark (Table 8.2) show consistent improvements of the BIMU model over SG and MU across all parallel corpora, except on the small CZ-EN (NC) corpus. We have also measured the 95% confidence intervals of the difference between the correlation coefficients of BIMU and SG, following the method described in Zou (2007). According to these values, BIMU significantly outperforms SG on RU-EN, and on French, Russian and Spanish NC corpora.[7]

Next, ignoring any language-specific factors, we would expect to observe a trend according to which the larger the corpus, the higher the correlation score. However, this is not what we find. Among the largest corpora, i.e. RU-EN, CZ-EN and FR-EN, the models trained on RU-EN perform surprisingly well, practically on par with the 23-times larger FR-EN corpus. Similarly, the quality of the embeddings trained on CZ-EN is generally lower than when trained on the 10 times smaller RU-EN corpus. One explanation for this might be different text composition of the corpora, with RU-EN matching the domain of the evaluation task better than the larger two corpora. Also, FR-EN is known to be noisy, containing web-crawled sen-

[7] We count those results in which the CI of the difference does not include 0.

					BiMu↔Sg	
Task	Corpus	Sg	Mu	BiMu	Diff	CI_{Diff}
SCWS	RU-EN	54.8	57.3	**59.5**	4.7	0.9–9.8
	CZ-EN	51.2	54.0	**55.3**	4.1	-0.6–8.8
	FR-EN	58.8	60.4	**60.5**	1.7	-2.6–5.9
	FR-EN (NC)	47.2	52.4	**54.3**	7.1	2.2–12.0
	RU-EN (NC)	47.3	**54.0**	**54.0**	6.7	0.6–12.8
	CZ-EN (NC)	47.7	**52.1**	51.9	4.2	-2.0–10.3
	DE-EN (NC)	48.5	52.9	**54.0**	5.5	-0.6–11.6
	ES-EN (NC)	47.2	53.2	**54.5**	7.3	1.1–13.3
Similarity	RU-EN	37.8	41.2	**46.3**		
	CZ-EN	39.5	36.9	**41.9**		
	FR-EN	**46.3**	42.0	43.5		
	FR-EN (NC)	17.9	26.0	**27.6**		
	RU-EN (NC)	19.3	27.3	**28.4**		
	CZ-EN (NC)	15.8	**26.6**	25.4		
	DE-EN (NC)	20.7	28.4	**30.8**		
	ES-EN (NC)	19.9	27.2	**31.2**		
Qvec	RU-EN	55.8	56.0	**56.5**		
	CZ-EN	**56.6**	56.5	55.9		
	FR-EN	57.5	57.1	**57.6**		
POS	RU-EN	**93.5**	93.2	93.3		
	CZ-EN	**94.0**	93.7	**94.0**		
	FR-EN	**94.1**	93.8	94.0		

Table 8.2: Results, per-row best in bold. Sg and Mu are trained on the English part of the parallel corpora. In BiMu↔Sg, we report the difference between BiMu and Sg, together with the 95% CI of that difference. The Similarity scores are averaged over 12 benchmarks described in subsection 8.6.1. For POS tagging, we report the accuracy.

tences that are not parallel or not natural language (Denkowski et al., 2012). Furthermore, language-dependent effects might be playing a role: For example, there are signs of Czech being the least helpful language among

Model (300-dim.)	SCWS
SG	65.0
MU	66.7
BIMU	69.0
Chen et al. (2014)	68.4
Neelakantan et al. (2014)	69.3
Li and Jurafsky (2015)	69.7

Table 8.3: Comparison to other works (reprinted), for the vocabulary of top-6000 words. Our models are trained on RU-EN, a much smaller corpus than those used in previous work.

those studied. But while there is evidence for that in all intrinsic tasks, the situation in POS tagging does not confirm this speculation. We have a closer look at the effect of language choice on the quality of word embeddings in section 8.8.2.

To gain some insight in how our models compare to the results previously reported in the literature, we relate our models to previously reported SCWS scores using 300-dimensional models in Table 8.3. Even though we train on a much smaller corpus than the previous works,[8] the BIMU model achieves a very competitive correlation score.

The results on Similarity benchmarks and Qvec largely confirm those on SCWS, despite the lack of sentential context which would allow to weight the contribution of different senses more accurately for the multi-sense models. Why, then, does simply averaging the MU and BIMU embeddings lead to better results than when using the SG embeddings? We hypothesize that the single-sense model tends to over-represent the dominant sense with its generic, one-vector-per-word representation, whereas the uniformly averaged embeddings yielded by the multi-sense models better encode the range of potential senses. Similar observations have been made in the context of selectional preference modeling of polysemous

[8]For example, Li and Jurafsky (2015) use the concatenation of Gigaword and Wikipedia with more than 5 billion words.

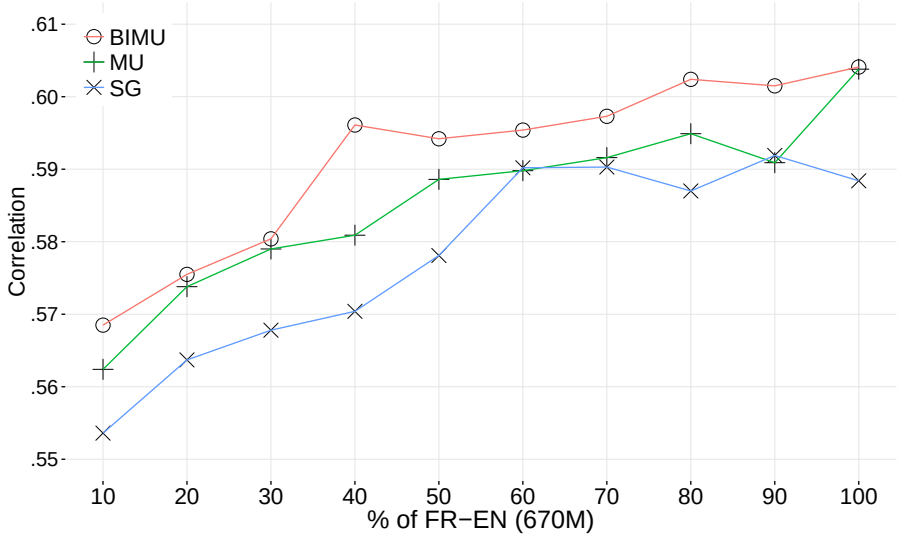


Figure 8.2: Effect of amount of data used in learning on the SCWS correlation scores.

verbs (Greenberg et al., 2015).

In POS tagging, the relationship between MU and BIMU models is similar as discussed above. Overall, however, neither of the multi-sense models outperforms the SG embeddings, which is somewhat surprising. A possible explanation is that the neural network tagger may be able to implicitly perform disambiguation on top of single-sense SG embeddings, similarly to what has been argued in Li and Jurafsky (2015). The tagging accuracies obtained with MU on CZ-EN and FR-EN are similar to the one obtained by Li and Jurafsky with their multi-sense model (93.8), while the accuracy of SG is more competitive in our case (around 94.0 compared to 92.5), although they use a larger corpus for training the word representations.

In all tasks, the addition of the bilingual component during training increases the accuracy of the encoder for most corpora, even though the bilingual information is not available during evaluation.

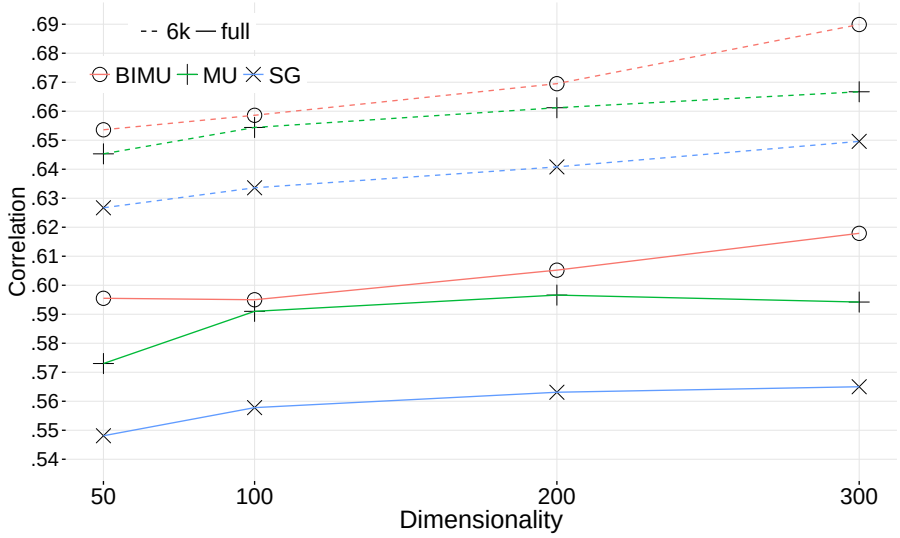


Figure 8.3: Effect of embedding dimensionality on the models trained on RU-EN and evaluated on SCWS with either the full vocabulary or the top-6000 words.

8.7.2 The amount of (parallel) data

Figure 8.2 displays how the semantic similarity as measured on SCWS evolves as a function of increasingly larger sub-samples from FR-EN, our largest parallel corpus. The BiMu embeddings show relatively stable improvements over Mu and especially over Sg embeddings. The same performance as that of Sg at 100% is achieved by Mu and BiMu sooner, using only around 40 or 50% of the corpus.

8.7.3 The dimensionality and frequent words

It is argued in Li and Jurafsky (2015) that often just increasing the dimensionality of the Sg model suffices to obtain better results than that of their multi-sense model. We look at the effect of dimensionality on semantic

similarity in Figure 8.3, and see that simply increasing the dimensionality of the SG model (to any of 100, 200 or 300 dimensions) is not sufficient to outperform the MU or BIMU models of the same dimensionality. When constraining the vocabulary to 6,000 most frequent words (the dashed lines in the figure), the representations obtain higher quality. This is understandable since more evidence is available during training for frequent words. We can see that the models, especially SG, benefit slightly more from the increased dimensionality when looking at these most frequent words. This is also according to expectations—frequent words need more representational capacity due to their complex semantic and syntactic behavior (Atkins and Rundell, 2008).

8.7.4 The number of senses

In our model, the number of senses k is a parameter, which we keep fixed to 3 throughout the empirical study. We comment here briefly on other choices, namely $k \in \{2, 4, 5\}$. We have found $k = 2$ to be a good choice on the RU-EN and FR-EN corpora (but not on CZ-EN), with an around 0.2-point improvement over $k = 3$ on SCWS and in POS tagging. With the larger values of k, the performance tends to degrade. For example, on RU-EN, the $k = 5$ score on SCWS is about 0.6 point below our default setting.

8.8 The effect of second language

In this section, we first have a look at the effect of varying the contribution of the first and the second language, and then how the choice of the second language and the language family affects the embedding quality and the outcome on the SCWS and in POS tagging.

8.8.1 The importance of bilingual signal

The degree of contribution of the second language l' during learning is affected by two different parameters. One is the parameter λ, controlling

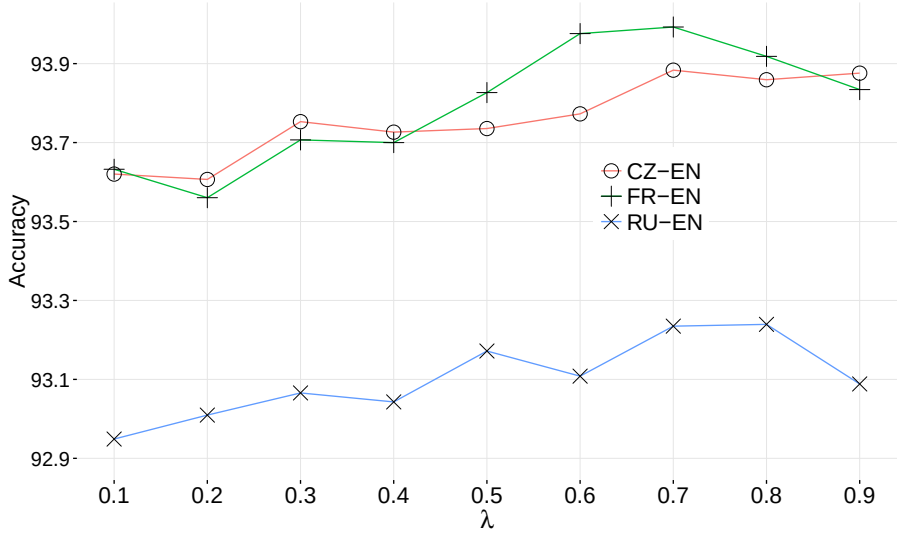


Figure 8.4: Controlling the bilingual signal: effect of varying the parameter λ for controlling the importance of second-language context (0.1-least important, 0.9-most important). The reported accuracies are measured on the POS tagging development set.

the trade-off between the importance of the first and the second language in the sense prediction part (encoder), the other is the value of m for controlling the size of the window around the second-language word that is affiliated to the pivot (equation (8.7)). The effect of the first parameter is shown graphically in Figure 8.4. It suggests that the context from the second language is useful in sense prediction, and that it should be weighted relatively heavily (around 0.7 and 0.8, depending on the language).

Regarding the role of the second parameter in sense disambiguation, the WSD literature has reported both smaller (more local) and larger (more topical) monolingual contexts to be useful, see e.g. Ide and Véronis (1998) for an overview. In Figure 8.5 we find that considering a very narrow context in the second language—the affiliated word only or a $m = 1$ window around it—performs the best, and that there is usually some deterio-

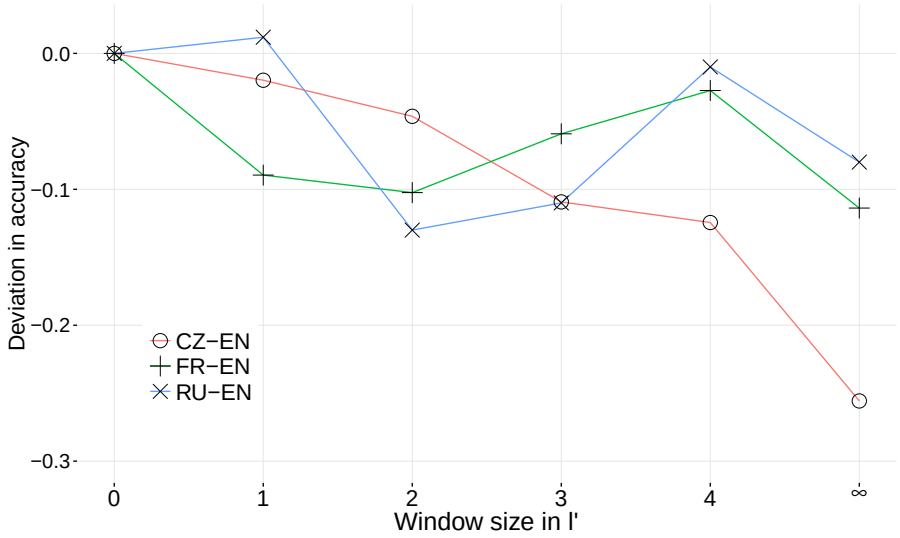


Figure 8.5: Controlling the bilingual signal: effect of second-language window size m on the accuracy. The reported accuracies are measured on the POS tagging development set.

ration when using a broader window. However, the negative effect on the accuracy is still relatively small, up to around -0.1 for the models using French and Russian as the second languages, and -0.25 for Czech when setting $m = \infty$ (the rightmost label on the x-axis). It is interesting that the infinite window size setting, which approximately corresponds to using sentence-only alignments[9], performs well also on the SCWS benchmark, and improves on the monolingual multi-sense baseline on all corpora. The SCWS results are shown in Table 8.4.

[9]Strictly speaking, the $m = \infty$ setting is not the same as using only sentence alignments since the entire sentence is taken as the context *only* when the affiliated word exists, but no second-language context is used when a word has no affiliated counterpart. In practice, though, we expect the difference in quality of word representations between the two views to be small.

Model	RU-EN	CZ-EN	FR-EN
MU	63.29	59.12	64.19
BIMU, $m = \infty$	**65.61**	**62.07**	**64.36**

Table 8.4: Comparison of SCWS correlation scores of BIMU trained with infinite l' window to the MU baseline (vocabulary of top-6000 words).

8.8.2 The effect of language choice

Throughout this chapter, we have used several languages in turn to study their effect on the sense-encoding component of our model. But because of the variation inherent to the parallel corpora representing these languages—most importantly the size, as well as the text and topic composition—we have been unable to draw any conclusions about *which* languages are the most beneficial for performing sense discrimination in English. In this section, we therefore examine the contribution of different languages in a controlled setting. We achieve this by running a small-scale study on a multi-parallel corpus, i.e. a corpus in which several languages are sentence-aligned to a single language, in our case English.

Before presenting the empirical study, we would like to briefly discuss what we can expect intuitively. Arguably, languages closely related to English should be less beneficial in improving the sense estimator for English than the more distant languages. For example, the corresponding polysemy of an English word in a language like French might be simply preserved, e.g. "interest" (English) and "intérêt" (French). Given "interest" in an English sentence, it is of little help to know that this word occurrence is translated as "intérêt" in French, since both can bear the financial and the wanting-to-know meaning. However, in another language like Slovene, this meaning distinction would be lexicalized with e.g. "zanimanje" (wanting-to-know) and "obresti" (financial). It could then present a useful source of information when disambiguating the English side. In general, we would expect the benefit to be greater with more distant languages, as there would be fewer situations with overlapping polysemy.

It is exactly this type of research questions that have kept busy also the researchers in word-sense disambiguation (WSD) (Ide, 2000; Resnik and Yarowsky, 1999). Ide (2000), for example, studies the degree to which translations lexicalize a source word differently. She finds that a word like "head" occurs in an English novel 65 times and with 4 different senses according to WordNet. It is lexicalized with 9 different words in Slovene and Romanian, with 6 words in Spanish, and 4 words in Czech. Although Ide also categorizes languages according to their relatedness (languages of the same genus vs. between genera, as well as different families (non- vs. Indoeuropean)), she does not discover any link between the language distance and the degree of lexicalization. This is unlike the previous study of Resnik and Yarowsky (1999), who found that monolingual sense distinctions can be captured well especially when the language family distance increases (non-Indoeuropean languages tend to lexicalize English sense distinctions more than Indoeuropean languages). In our case, we would like to see which of the findings can be confirmed in our own experiments with representation learning.

We choose the Bible as our multi-parallel corpus. It is readily available and translated in many languages, including those we have studied in the preceding sections of this chapter. We use the translations available from `http://homepages.inf.ed.ac.uk/s0787820/bible/`. This resource, discussed in detail in Christodouloupoulos and Steedman (2014), includes 100 translations of the Bible in total, most of them into non-Indoeuropean languages. For around half of the languages, full translations of the Bible are available; these amount to around 31 thousand sentences or verses (per language). The size of the corpus in words varies between approximately 690 and 920 thousand words. We use the following languages in our experiments:

- 9 Indoeuropean languages belonging to six different genera or subfamilies: Italic (French, Spanish), Slavic (Russian, Czech, Slovene), Baltic (Lithuanian), Germanic (German), Greek (Greek) and Albanian (Albanian);

Task	Language	Correlation	Indo-european: yes / no
SCWS	German	0.50	
	Xhosa	0.50	
	Tagalog	0.49	
	Slovene	0.49	
	French	0.49	
	Arabic	0.49	
	Greek	0.49	
	Somali	0.49	
	Hebrew	0.48	
	Russian	0.48	
	Czech	0.48	
	Lithuanian	0.48	
	Albanian	0.48	
	Vietnamese	0.48	
	Spanish	0.47	
	Telugu	0.46	

Table 8.5: SCWS benchmark results with bilingually trained word embeddings. English is used as the first language, and one of the listed languages as the second language included in the sense estimation step. Blue denotes Indoeuropean languages; red denotes non-Indoeuropean languages.

- 7 non-Indoeuropean languages from six different genera: Semitic (Arabic, Hebrew), Atlantic-Congo (Xhosa), Malayo-Polynesian (Tagalog), Cushitic (Somali), Mon-Khmer (Vietnamese) and South-Central (Telugu).

Just as with the previously used corpora, we prepare the Bible corpora by first tokenizing and then aligning them on the word level. The minimum frequency of words included in the word-representation vocabulary is 5.

The effect of the choice of the second language on the quality of representations as measured intrinsically is shown in Table 8.5. The range of

correlation scores from highest to lowest is small, which indicates that the language choice does not play an important role on the quality of word representations (when evaluating on this particular dataset). There is also no clear pattern depending on the language family, and at the same time more and less distant languages can occur at a similar place in the ranking: German as a closely related language and Xhosa as a distant language both perform similarly.

We evaluate the embeddings trained with these different languages also in the POS tagging task. The results are shown in Figure 8.6. The languages are ranked according to the accuracy and depending on the language family. The range of accuracies obtained is somewhat larger (max. around 0.5 point) than in the case of SCWS, but similarly to the previous experiment, it is hard to discern any strong tendency. The Indoeuropean languages perform on average just as well, if not slightly better (at a difference of around 0.1 in median score) than the non-Indoeuropean ones.

We can conclude from these experiments that the language relatedness does not influence the quality of our sense encoder and the obtained word representations in any predictable way. Compared to the WSD works introduced above, our findings are closer to those of Ide (2000), who did not find any clear relationship either. Despite the outcome of our study, we can not state that language choice does not affect the end quality of word representations *in general*. Our analysis is based on a corpus which is small according to representation learning standards; it is entirely possible that a study involving a larger corpus would yield different results. Unfortunately, not many corpora suitable for such a study exist.[10] Secondly, the effect of language choice is not measured directly, as in a WSD set-up, but only through the effect of the sense estimation component on the embedding learning part. Furthermore, the representations are aggregated at test time using a weighted average, which might smear out some differences.

[10]One candidate is Europarl (Koehn, 2005), but it offers a narrower range of languages, with varying amounts of available data. A study could still be performed using the common intersection of a selection of languages.

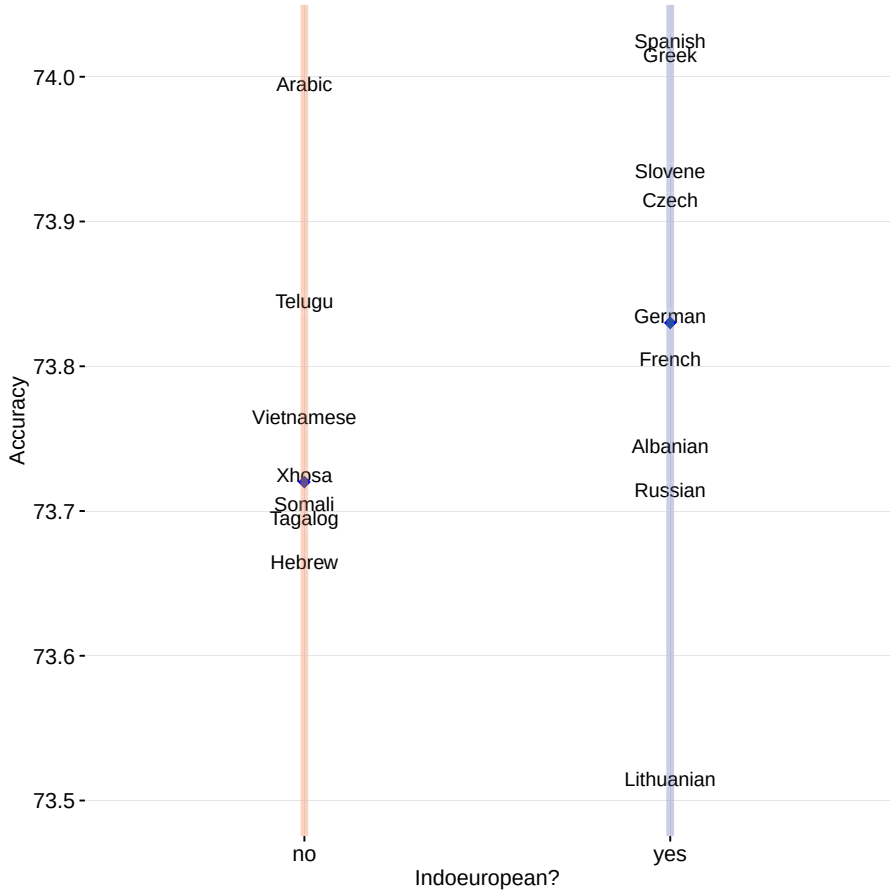


Figure 8.6: POS tagging accuracies with bilingually trained word embeddings. English is used as the first language, and one of the listed languages as the second language (included in the sense estimation step). We distinguish between Indoeuropean and non-Indoeuropean languages, with the median value in each category denoted as a blue point.

8.9 Additional related work

In this section, we provide an overview of the related work, some of it not yet mentioned in the previous sections.

Multi-sense models. One line of research has dealt with sense induction as a separate, clustering problem that is followed by an embedding learning component (Huang et al., 2012; Reisinger and Mooney, 2010). In another, the sense assignment and the embeddings are trained jointly, just like in our case (Neelakantan et al., 2014; Tian et al., 2014; Li and Jurafsky, 2015; Bartunov et al., 2015). The two approaches most closely related to ours are by Neelakantan et al. (2014) and Li and Jurafsky (2015). Neelakantan et al. (2014) propose an extension of Skip-Gram (Mikolov et al., 2013a) by introducing sense-specific parameters together with the k-means-inspired "centroid" vectors that keep track of the contexts in which word senses have occurred. They explore two model variants, one in which the number of senses is the same for all words, and another in which a threshold value determines the number of senses for each word. The results comparing the two variants are inconclusive, with the advantage of the dynamic variant being virtually nonexistent. In our work, we use the static approach. Whenever there is evidence for less senses than the number of available sense vectors, this is unlikely to be a serious issue as the learning would concentrate on some of the senses, and these would then be the preferred predictions also at test time. Li and Jurafsky (2015) build upon the work of Neelakantan et al. with a more principled method for introducing new senses using the Chinese Restaurant Processes (CRP). Our experiments confirm the findings of Neelakantan et al. that multi-sense embeddings improve Skip-gram embeddings on intrinsic tasks, as well as those of Li and Jurafsky, who find that multi-sense embeddings offer little benefit to the neural network learner on extrinsic tasks. Our discrete-autoencoding method when viewed without the bilingual part in the encoder has a lot in common with their methods.

Multilingual models. The research on using multilingual information in the learning of *multi-sense* embedding models is scarce. Guo et al. (2014) perform a sense induction step based on clustering translations prior to learning word embeddings. Once the translations are clustered, they are mapped to a source corpus using WSD heuristics, after which a recurrent neural network is trained to obtain sense-specific representations. Unlike in our work, the sense induction and embedding learning components are entirely separated, without a possibility for one to influence another. In a similar vein, Bansal et al. (2012) use bilingual corpora to perform soft word clustering, extending the previous work on the monolingual case of Lin and Wu (2009). *Single-sense* representations in the multilingual context have been studied more extensively (Lu et al., 2015; Coulmance et al., 2015; Faruqui and Dyer, 2014b; Hill et al., 2014a; Zhang et al., 2014; Faruqui and Dyer, 2013; Zou et al., 2013), with a goal of bringing the representations in the shared semantic space. A related line of work concerns the crosslingual setting, where one tries to leverage training data in one language to build models for typically lower-resource languages (Hermann and Blunsom, 2014; Gouws et al., 2014; Chandar A P et al., 2014; Soyer et al., 2014; Klementiev et al., 2012; Täckström et al., 2012).

The recent works of Kawakami and Dyer (2016) and Nalisnick and Ravi (2015) are also of interest. The latter work on the infinite Skip-Gram model in which the embedding dimensionality is stochastic is relevant since it demonstrates that their embeddings exploit different dimensions to encode different word meanings. Just like us, Kawakami and Dyer (2016) use bilingual supervision, but in a more complex LSTM network that is trained to predict word translations. Although they do not represent different word senses separately, their method produces representations that depend on the context. In our work, the second-language signal is introduced only in the sense prediction component and is flexible—it can be defined in various ways and can be obtained from sentence-only alignments as a special case.

8.10 Conclusion and future work

We have presented a method for learning multi-sense embeddings that performs sense estimation and context prediction jointly. Both mono- and bilingual information is used in the sense prediction during training. We have explored the model performance on a variety of tasks, showing that the bilingual signal improves the sense predictor, even though the crosslingual information is not available at test time. In this way, we are able to obtain word representations that are of better quality than the monolingually-trained multi-sense representations, and that outperform the Skip-Gram embeddings on intrinsic tasks. We have analyzed the model performance under several conditions, namely varying dimensionality, vocabulary size, amount of data, and size of the second-language context. For the latter parameter, we find that bilingual information is useful even when using the entire sentence as context, suggesting that sentence-only alignment might be sufficient in certain situations. We have also examined the effect of language choice on the quality of embeddings by using a multi-parallel corpus. We have not found any clear relationship between the language or its family and the outcome in evaluation tasks in which we use the bilingually trained word embeddings.

There are many research areas we did not address in our work, but are worth exploring. One is to create a truly multilingual set-up in which several languages contribute to the crosslingual signal at the same time. Our formulation of the sense encoder allows such an extension by a straightforward modification, in which several languages combine linearly in the encoder. In that case, it would be necessary that the crosslingual signal comes from word representations sharing the same embedding space. Such representations can be estimated with methods described in Coulmance et al. (2015), Guo et al. (2016) and Luong et al. (2015), inter alia. Secondly, while we have used various languages in enhancing the word embeddings for English, it would make sense to reverse the situation and use English (or other languages) to train embeddings for a language other than English. In that case, the set of the evaluation tasks might need to be reconsidered

due to the scarcity of annotated resources in many non-English languages. Finally, the extrinsic performance of multi-sense embeddings in general should be carefully examined since the advantage is not clear based on our findings and that of Li and Jurafsky (2015). Preferably, a larger array of extrinsic tasks would be considered. Additionally, it would be interesting to directly compare the accuracy of a POS tagging model implemented as a neural network with other non-neural (sequence) classifiers.

PART V

Conclusion

CHAPTER 9

Summary and conclusions

In this thesis, we focused on word representations and proposed several methods that build upon existing approaches. We investigated their properties and effectiveness in a variety of settings. We have addressed three major themes: The first and the most specific one was the application of human-built word representations in syntactic parsing (Part II); the second (Part III) concerned the role of syntax in models of word representations; and lastly, we looked at the role of multilingual learning in the representation models capable of explaining polysemy (Part IV).

In Part I, we started off with a presentation of theoretical background on word representations. We introduced distributional lexical semantics and motivated the use of word representations in natural language processing through the concepts of lexical sparseness and generalization. We distinguished between two ways of evaluating the word representations: intrinsic, which aims to measure how successfully the word representations capture human-elicited word similarity; and extrinsic, which involves the integration of representations in concrete language processing systems. We have also shown that representation learning can be seen as a constituent part of the broader semi-supervised machine learning. In Chapter 3, we gave an overview of all the representation methods we worked

with: concept classes, word clusters, latent-variable models and word embeddings. In Chapter 4, we went on to describe the evaluation tasks that included several semantic similarity benchmarks and models for part-of-speech tagging, named entity recognition, syntactic parsing, and identification of semantic frames.

In Part II, we empirically investigated the lexical sparseness phenomenon in the Alpino parser for Dutch. We extended its component for modeling bilexical preferences with wordnet classes. Even though this led to improvements in certain parsing outcomes, our method also introduced in fact just as many incorrect parses. The main finding—which also answers the research question we formulated in the introduction (**Q1**)—was thus mostly negative: The generalization through concept classes did not lead to improved overall parsing accuracy. We conjectured that this could have happened because we did not build the disambiguation component with generalization properties from scratch, but based on Alpino's existing disambiguation component, which already tackled lexical sparseness issues to a large degree. We also found that many errors made by the parser could not be solved by means of generalization alone. The generalization effects that interested us could also be studied in data-driven parsers which are in general more heavily lexicalized than Alpino (cf. Plank (2011)); in those, the overall improvement in parsing accuracy might be larger.

While the human-crafted concept representations have several attractive properties like interpretability, sense specificity and hierarchical organization, we found that many word types were not covered by the resource. An additional problem of wordnets might be their excessive sense differentiation. Also, sense disambiguation of word occurrences can be challenging and error-prone. For these reasons, we decided to explore distributional word representations in the remaining parts of the thesis.

In Chapters 6 and 7 of Part III, we addressed the research question **Q2** that concerned the usefulness of syntactic context in learning of word representations. We focused on incorporating different kinds of syntactic information in Brown clustering and Hidden Markov models. The extension

of Brown clustering with a dependency language model proved successful for Dutch. We obtained clusters that were closer to human-built classes of semantically related words than those obtained with bigram-based Brown clustering. Also, as we showed later, they performed well as features in named entity recognition for Dutch. We then proceeded with a more complex model by relying on the following reasoning. First, Brown clustering makes a simplifying assumption that a word may only belong to one cluster, and in this way fails to account for the polysemy of words. Second, certain experiments in cluster induction from Chapter 6 suggested that, instead of relying solely on dependency structures without labels, it might be preferable to refine the word contexts by including the information about dependency labels. We implemented this idea in Chapter 7, in which we studied syntactic variants of Hidden Markov models. Our approach included a model in which syntactic functions were added as additional observed variables. Based on the extrinsic evaluation for Dutch and English, the findings were twofold. First, we established that the addition of syntactic functions in most cases yielded representations of better quality than those obtained from sequential and unlabeled-tree HMM models, thus supporting our reasoning about the need to represent syntactically different contexts with separate model parameters. Secondly, somewhat disappointingly, the quality of word representations obtained from those latent-variable models was inferior to that of the baseline systems that included Brown clusters and word embeddings, especially for English. Compared to Brown clusters, the most apparent benefit of HMMs is that they model a soft relationship between words and states, and provide context-sensitive representations. In practice though, the advantage over Brown clusters was not always evident. Furthermore, HMM word representations are computationally more expensive to obtain and need a more intensive exploration of the parameter space to work well. The upshot of our exploration in Part III is therefore that both unlabeled and labeled syntactic structures can lead to a more precise and informative definition of context than plain sequences. However, since the benefit was not observed univer-

sally, factors like the language and the model complexity should be taken into account. In future, one would then need to examine more closely how the language choice influences the effect of syntactic context on the quality of word representations (e.g. what is it precisely that makes the syntactic contexts more beneficial for Dutch than for English?). Another avenue for future research is to study the discrimination between context types in other word representation frameworks (cf. Wang et al. (2015) and Ling et al. (2015)). Also, since both Brown clustering and the latent-variable models rely on automatically acquired syntactic analyses, a natural area for exploration is the role of parsing mistakes on the quality of learned representations.

In the last part of the thesis, we studied neural word embeddings. The models we proposed are capable of distinguishing between word senses, similarly to HMM representations from the previous chapter. Using an autoencoding approach, we estimated iteratively and in an unsupervised manner the sense distributions for words and the embedding component. The emphasis in this part was to study the effect of crosslingual (i.e. second-language) contexts obtained from parallel corpora on the sense encoder for the first language. By evaluating in a variety of semantic similarity tasks, we established that the second-language signal often led to improvements compared to the model that only had access to monolingual contexts. This means that we can answer our third research question (**Q3**) positively: The bilingual corpora can offer the supervision that leads to improved multi-sense representations in the embedding framework. On a related note, we also found that multi-sense representations outperformed generic, single-sense representations in the intrinsic tasks. Surprisingly, this was not the case in our extrinsic task (POS tagging). Therefore, some questions that remain are whether multi-sense representations can be useful in other downstream tasks, and specifically, how and in what sort of predictor they should be included to be beneficial. A related question is whether information about polysemy can be effectively encoded in generic representations and then recovered as needed (cf. Nalisnick and Ravi (2015)). Also,

while we have not found any strong link between the choice of the language that provided the supervisory signal and the quality of the sense encoding component (or the resulting embeddings), further experiments would be needed to arrive at a more definite conclusion.

Finally, although we have worked in this thesis in several representation frameworks and addressed different research questions that are important for understanding word representations, the scope of investigation was throughout limited to words only. One question that is pertinent to any representation method is how do the word representations and our knowledge about them convey to the understanding of larger units such as phrases, sentences and documents (Manning, 2016; Kiros et al., 2015; Le and Mikolov, 2014; Grave et al., 2014; Blacoe and Lapata, 2012).

Appendices

APPENDIX A

Dependency Brown clustering objective

A.1 Simplifying the objective function

Our dependency Brown clustering objective is based on a class-based dependency language model, defined in section 6.3 as:

$$L(S^T; \sigma, T) = \prod_{i=1}^{m} p(w_i|\sigma(w_i))p(\sigma(w_i)|\sigma(w_{\pi(i)}), T), \tag{A.1}$$

where S is a lexicalization of the tree T with a word ordering $\langle w_i \rangle_{i=1}^{m}, w_i \in V$. The functions π and σ are the parent- and cluster-assigning functions, respectively.

The objective can be rewritten in terms of mutual information (Brown et al., 1992), which is the criterion actually used for optimizing the clustering function. From now on, we will work with a normalized binary logarithm of the probability, and we will omit the symbol T from the conditioning in the right-hand side of equations. The rewriting steps follow the

exposition in the lecture notes[1] of Jan Hajič , but we use the dependency instead of the bigram class-based language model. Slightly different rewrite sequences can be found in Brown et al. (1992) and Liang (2005).

$$L(S^T, \sigma, T) = \frac{1}{m}\sum_{i=1}^{m} \log(p(w_i|\sigma(w_i))p(\sigma(w_i)|\sigma(w_{\pi(i)}))) \tag{A.2}$$

$$= \frac{1}{m}\sum_{i=1}^{m} \log\left(p(w_i|\sigma(w_i))\underline{p(\sigma(w_i))}\frac{p(\sigma(w_i)|\sigma(w_{\pi(i)}))}{\underline{p(\sigma(w_i))}}\right) \tag{A.3}$$

$$= \frac{1}{m}\sum_{i=1}^{m} \log\left(\underline{p(w_i, \sigma(w_i))}\frac{p(\sigma(w_i)|\sigma(w_{\pi(i)}))}{p(\sigma(w_i))}\right) \tag{A.4}$$

$$= \frac{1}{m}\sum_{i=1}^{m} \log(\underline{p(w_i)}) + \frac{1}{m}\sum_{i=1}^{m} \log\left(\frac{p(\sigma(w_i)|\sigma(w_{\pi(i)}))}{p(\sigma(w_i))}\right) \tag{A.5}$$

$$= -H(W) + \frac{1}{m}\sum_{i=1}^{m} \log\left(\frac{p(\sigma(w_i)|\sigma(w_{\pi(i)}))\underline{p(\sigma(w_{\pi(i)}))}}{\underline{p(\sigma(w_{\pi(i)}))}p(\sigma(w_i))}\right) \tag{A.6}$$

$$= -H(W) + \frac{1}{m}\sum_{i=1}^{m} \log\left(\frac{\underline{p(\sigma(w_i), \sigma(w_{\pi(i)}))}}{p(\sigma(w_{\pi(i)}))p(\sigma(w_i))}\right) \tag{A.7}$$

$$= -H(W) + \sum_{d,e\in C} p(d,e)\log\left(\frac{p(d,e)}{p(d)p(e)}\right) \tag{A.8}$$

$$= -H(W) + I(D, E) \tag{A.9}$$

In each step, we use dashed underlining to mark the new or the changed term. The steps rewrite equation (A.2) to arrive at the entropy of the word distribution (first term in equations (A.5)–(A.9)) and the mutual information I over the variables D and E, representing the classes of words in a

[1] `http://www.cs.jhu.edu/~hajic/courses/cs465/cs46511/ppframe.htm`.

child-parent relationship (second term in equations (A.8)–(A.9)).

We are interested in finding the clustering function σ that maximizes the objective L:

$$\sigma' = \arg\max_{\sigma} L(\sigma, T). \tag{A.10}$$

Since the entropy of the word distribution in equation (A.9) does not depend on the clustering function σ, we are seeking to optimizing $I(D, E)$ only.

A naïve implementation would run in $O(|V|^5)$ time, namely, for each of $O(|V|)$ word types and for each of $O(|V|^2)$ possible cluster pairs to merge it would need to evaluate the resulting clustering quality, which would sum over $O(|V|^2)$ terms. This makes the algorithm far too impractical to run for typical vocabularies whose size is in the order of tens or hundreds of thousand of words. With several optimization and tricks, the running time can be brought down to $O(k^2|V|)$, where k is the number of clusters chosen prior to clustering. We refer the reader to Liang (2005) for the details.

APPENDIX B

Sum-product message passing

B.1 Background

In a Hidden Markov tree model with an additional observed variable, like the one we have introduced in Chapter 7, we work on an undirected tree with nodes V and edges E, in which each node $s \in V$ except the root has associated discrete random observed variables—in our case, words $w_s \in W$ and syntactic functions $r_s \in R$—as well as a discrete random hidden variable $c_s \in C$. We are given the data as $\{(w^1, r^1), \ldots, (w^M, r^M)\} \subseteq W \times R$, where M is the size of the dataset in sentences, and w_s^1, for example, is then the word at some node in the first sentence. In this chapter, we omit the conditioning on the tree to avoid overcrowded notation, and simplify by assuming that this information is already encoded by r.

The model consists of transition and emission parameters $\theta = (T, O)$. Once the parameters are learned, we can use them to obtain word representations as described in section 7.4.2. Intuitively, a good θ is one that fits the data well, i.e. maximizes its likelihood. In unsupervised learning of HMMs, we can use EM to find maximum likelihood solutions.

When working in the log domain, the learning objective is to minimize the negative log-likelihood of the data:

$$L(\theta) = -\log p(W|R;\theta) \tag{B.1}$$

$$= -\frac{1}{M}\sum_{m=1}^{M}\log\sum_{c^m} p(W = w^m, C = c^m|R = r^m;\theta). \tag{B.2}$$

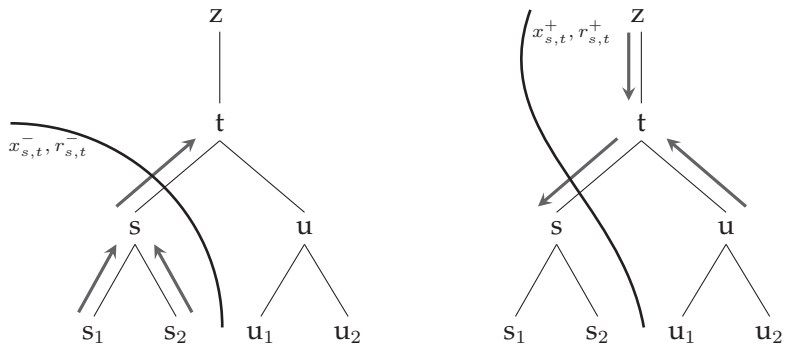
Because of the hidden variables, the likelihood contains a sum over all possible hidden structures c, which makes it hard to compute. We therefore use EM and the inference procedure described next, which will allow us to fill in the missing states using the knowledge of observed data and current parameters.

The goal in this chapter is to show how to compute the posterior probabilities that we need in order to collect the pseudo-counts (equation (7.3)), used in the maximization step of EM to estimate the (new) maximum likelihood parameters θ' at each iteration. We are concerned with computing two different types of posterior probabilities. One is the state posterior distribution $p(c_s|w, r)$ over the states in a certain node given the observations; another is the transition posterior distribution $p(c_s, c_{\pi(s)}|w, r)$, which are the probabilities of each transition between a node and its parent given the observations. All the quantities needed to calculate these posteriors can be obtained with the sum-product message passing algorithm.

B.2 Message passing

We now describe the sum-product message passing, or the belief propagation algorithm[1] (Pearl, 1988), which can be used to compute the posterior

[1]Note that some authors make a distinction between the two, e.g. Murphy (2012), depending on whether the top-down messages depend on the bottom-up messages along the same edge or are independent. Our presentation follows the sum-product version, in which, like in the forward-backward recursions (Stratonovich, 1960; Rabiner, 1989), the top-down messages are computed independently of the bottom-up ones.



(a) Bottom-up phase. (b) Top-down phase.

Figure B.1: Message passing on a tree, rooted at z. The direction of message passing is shown with arrows. The visible evidence is denoted with $w_{s,t}$ and $r_{s,t}$, and by using superscript $-$ and $+$ to indicate the downstream and the upstream side of the tree. The illustration is based on Murphy (2012).

probabilities of the hidden states in graphical models in general. In describing the algorithm for tree HMMs, we follow Murphy (2012), with a difference that we have an extra observed variable accounting for the syntactic functions, which in our model modulates both the hidden, "semantic class" variable and the observed word forms. The algorithm will allow us to find posterior marginals $p(c_s|w, r)$ for all nodes $s \in V$ in the tree, and transition marginals $p(c_s, c_{\pi(s)}|w, r)$ for all edges $(s, \pi(s)) \in E$, where w and r stand for the full word and syntactic function observations.

In message passing, we can in principle pick any of the tree nodes and designate it as the root, and we can also identify all the leaves of the tree by orienting all the edges away from the root. In our case, however, a dependency tree gives us an already predefined root node, which is also special in that it is not associated with any observed variable. An example tree structure is shown in Figure B.1. The method consists of propagating messages from the leaves towards the root in a bottom-up phase, as in B.1a,

and from the root towards the leaves in a top-down phase, as in B.1b. A message m can be seen as a real-valued function passed along edges and between nodes. It holds a summary of the evidence for the subtree from which it originates. A node can send a message to its parent once it has received all messages from its children. The message passing is performed recursively until messages have been propagated along every edge. We will use an auxiliary quantity, called belief, to represent the local evidence at a node.

Our Hidden Markov tree model with syntactic functions can be factored as

$$p(w, c|r) = \prod_{s \in V} \psi_s(c_s, w_s, r_s) \prod_{(s,\pi(s)) \in E} \psi_{s,\pi(s)}(c_s, c_t, r_s), \tag{B.3}$$

where $\psi_s(c_s, w_s, r_s) = p(w_s|c_s, r_s)$ is the *potential* of the local evidence (emission) at node s, and $\psi_{s,\pi(s)}(c_s, c_{\pi(s)}, r_s) = p(c_{\pi(s)}|c_s, r_s)$ is the edge (transition) potential at $(s, \pi(s))$. We use the "potential" notation to conform with the literature (Murphy, 2012; Barber, 2012; Wainwright and Jordan, 2008).

We start by picking a node from Figure B.1a, say t, and compute its belief state to hold the evidence encountered at or or below t:

$$\text{bel}_t^-(c_t) \triangleq p(c_t, w_t^-|r_t^-) = \psi_t(c_t, w_t, r_t) \prod_{\substack{t' \in B(t) \\ \setminus \{\pi(t)\}}} m_{t' \to t}^-(c_t), \tag{B.4}$$

where $B(t)$ is the set of immediate neighbors of t, i.e. the children and the parent: $\{t' \in V|(t, t') \in E\}$. In words, the bottom-up belief at node t is the product of the local emission potential and the incoming messages from the children of t. It can be interpreted as the probability of being in a certain state at the node t and having observed all the words given the syntactic functions on the downstream.

The computation of messages, which we show next, itself relies on the

downstream belief states. It is in this sense that the algorithm is recursive:

$$m^{-}_{s\to t}(c_t) = \sum_{i}^{N} \psi_{s,t}(c_s = i, c_t, r_s)\text{bel}^{-}_{s}(c_s = i), \tag{B.5}$$

where we sum over all N hidden variable elements. As we can see, the belief lower in the tree (at s) gets converted to belief higher in the tree (at t) by including the edge potential at (s,t), as well as the local node potential at t.

Once the upward phase reaches the root z, all nodes have been seen, so the local belief state can be computed:

$$\text{bel}^{-}_{z}(c_z) \triangleq p(c_z, w^{-}_{t}|r^{-}_{t}) = \prod_{z'\in B(z)} m^{-}_{z'\to z}(c_z). \tag{B.6}$$

Because in our model we do not associate any emission potential with the root node z and since the root does not have a parent, the only belief-update information is the product of incoming messages from its children. Since we have seen all the evidence in the tree, the belief state of the root can be interpreted as the probability of the observed tree, $p(w^{-}_{t}|r^{-}_{t}) = p(w|r)$.

In the second phase, the messages are passed from the root towards the leaves. In the computation of the top-down message from t to s, we assume, by recursion, that the top-down messages further upstream have already been calculated:

$$m^{+}_{t\to s}(c_s) \triangleq p(w^{+}_{s,t}|c_s, r^{+}_{s,t}) =$$
$$\sum_{i}^{N} \underline{\psi_{s,t}(c_s, c_t = i, r_s)}\underline{\psi_t(c_t = i, w_t, r_t)}\underline{m^{+}_{\pi(t)\to t}(c_t = i)}\prod_{\substack{t'\in B(t)\\ \setminus\{\pi(t),s\}}}\underline{m^{-}_{t'\to t}(c_t = i)}. \tag{B.7}$$

The top-down message represents the probability of observing all up-

stream words given the observed upstream syntactic functions and being in a certain state at the node s. The message holds all the information from the upstream side of the edge (s, t), including all the edge potentials and the node potentials, the top-down message from t's parent as well as the bottom-up messages to t originating from children other than s. Working like this through the tree, we obtain the top-down messages for all edges in the tree. These, together with the bottom-up node beliefs, make it possible to compute the state posteriors:

$$p(c_s|w, r) = \frac{p(c_s, w|r)}{p(w|r)} = \frac{\text{bel}_s^-(c_s) m_{t\to s}^+(c_s)}{p(w|r)}, \tag{B.8}$$

and the transition posteriors:

$$p(c_s, c_t|w, r) = \frac{1}{p(w|r)} \text{bel}_s^-(c_s) \psi_{s,t}(c_s, c_t, r_s) \psi_t(c_t, w_t, r_t) m_{z\to t}^+(c_t) \prod_{\substack{t' \in B(t) \\ \setminus \{\pi(t), s\}}} m_{t'\to t}^-(c_t). \tag{B.9}$$

The running time of the message passing algorithm is linear in the number of tree nodes $|V|$ and quadratic in the number of hidden states N.

APPENDIX C

Individual semantic similarity results from Chapter 8

Benchmark	SG	MU	BIMU
WS-353	47.85	47.83	53.59
WS-353-SIM	56.56	56.34	63.27
WS-353-REL	42.31	44.93	48.26
MC-30	53.24	48.46	61.55
RG-65	43.62	46.95	53.79
Rare-Word	37.28	38.04	40.96
MEN	38.93	47.71	51.23
MTurk-287	41.99	52.18	59.06
MTurk-771	39.01	41.01	45.18
YP-130	8.6	16.14	18.59
SimLex-999	19.04	23.04	23.04
Verb-143	25.37	31.26	37.18
Average	37.82	41.16	46.31

Table C.1: Individual results: RU-EN.

Benchmark	SG	MU	BIMU
WS-353	48.04	44.23	48.92
WS-353-SIM	55.23	54.84	58.56
WS-353-REL	41.6	36.2	37.28
MC-30	41.28	43.15	45.82
RG-65	40.94	43.33	43.62
Rare-Word	42.29	40.83	40.34
MEN	46.21	47.92	51.69
MTurk-287	48.16	47.92	54.17
MTurk-771	42.22	45.09	43.27
YP-130	27.48	19.48	32.14
SimLex-999	15.88	13.95	15.76
Verb-143	25.33	6.29	31.35
Average	39.56	36.94	41.91

Table C.2: Individual results: CZ-EN.

Benchmark	SG	MU	BIMU
WS-353	51.91	46.73	48.85
WS-353-SIM	64.1	58.65	62.38
WS-353-REL	42.65	38.09	40.3
MC-30	53.55	46.93	43.55
RG-65	52.63	44.27	44.24
Rare-Word	43.31	39.54	42.32
MEN	52.58	51.68	51.11
MTurk-287	59.03	59.42	60.12
MTurk-771	47.91	45.59	45.99
YP-130	31.56	31.22	36.6
SimLex-999	26.92	24.59	25.24
Verb-143	29.6	16.89	21.77
Average	46.31	41.97	43.54

Table C.3: Individual results: FR-EN.

Benchmark	SG	MU	BIMU
WS-353	29.19	37.82	41.94
WS-353-SIM	43.4	48.03	53.92
WS-353-REL	23.44	35.43	36.52
Rare-Word	22.38	25.24	34.46
MEN	24.14	17.54	22.85
MTurk-287	8.92	29.34	24.85
MTurk-771	26.04	30.43	32.42
YP-130	-6.62	15.43	27.04
SimLex-999	7.47	7.71	10.36
Verb-143	20.17	25	27.31
Average	19.85	27.20	31.17

Table C.4: Individual results: ES-EN (NC).

Benchmark	SG	MU	BIMU
	News-de		
WS-353	32.12	42.18	46.67
WS-353-SIM	50.45	49.79	54.71
WS-353-REL	23.74	38.4	42.95
Rare-Word	18.56	30.31	26.65
MEN	23.14	20.73	17.89
MTurk-287	4.68	27.8	29.82
MTurk-771	22.44	30.64	35.66
YP-130	5.73	9.79	16.73
SimLex-999	8.13	9.49	11.57
Verb-143	17.87	25.1	25.59
Average	20.69	28.42	30.82

Table C.5: Individual results: DE-EN (NC).

Benchmark	SG	MU	BIMU
WS-353	31.93	38.43	37.77
WS-353-SIM	47.98	44.92	47.23
WS-353-REL	25.5	33.26	30.08
Rare-Word	19.54	22.89	33.8
MEN	25.7	21.37	24.17
MTurk-287	-0.91	29.72	24.49
MTurk-771	21.61	26.07	35.46
YP-130	-2.73	17.57	16.55
SimLex-999	5.57	7.41	8.77
Verb-143	18.88	31.63	25.33
Average	19.31	27.33	28.37

Table C.6: Individual results: RU-EN (NC).

Benchmark	SG	MU	BIMU
WS-353	27.46	36	39.8
WS-353-SIM	44.22	48.29	49.6
WS-353-REL	20.8	31.84	35.2
Rare-Word	9.98	25.64	24.98
MEN	22.03	18.72	20.32
MTurk-287	-8.31	28.16	10.4
MTurk-771	17.55	33.6	30.41
YP-130	0.9	4.78	18.57
SimLex-999	2.78	8.88	6.46
Verb-143	20.92	30.42	18.31
Average	15.83	26.63	25.41

Table C.7: Individual results: CZ-EN (NC).

Benchmark	SG	MU	BIMU
WS-353	30.91	37.32	40.26
WS-353-SIM	44.78	46.06	49.42
WS-353-REL	25.15	33.88	34.09
Rare-Word	18.97	31.05	33.93
MEN	21	20.49	21.4
MTurk-287	-2.48	21.05	26.21
MTurk-771	19.93	32.04	35.71
YP-130	-2.86	15	11.05
SimLex-999	6.59	7.31	5.95
Verb-143	16.78	15.78	17.7
Average	17.88	26.00	27.57

Table C.8: Individual results: FR-EN (NC).

Bibliographical abbreviations

We use the following abbreviations for the proceedings of conferences and workshops:
AAAI → Conference on Artificial Intelligence
ACL → Conference of the Association for Computational Linguistics
ANLC → Conference on Applied Natural Language Processing
CIKM → ACM International Conference on Conference on Information and Knowledge Management
COLING → International Conference on Computational Linguistics
CoNLL → Conference on Computational Natural Language Learning
EACL → Conference of the European Chapter of the Association for Computational Linguistics
EMNLP → Conference on Empirical Methods in Natural Language Processing
GWC → Global WordNet Conference
HLT → Conference on Human Language Technology
ICASSP → International Conference on Acoustics, Speech, and Signal Processing
ICLR → International Conference on Learning Representations
ICML → International Conference on Machine learning
IJCNLP → International Joint Conference on Natural Language Processing
IWPT → International Conference on Parsing Technologies

LREC → Language Resources and Evaluation Conference
NAACL → Conference of the North American Chapter of the Association for Computational Linguistics
NIPS → Neural Information Processing Systems Conference
NLPLING → Workshop NLP and Linguistics
NODALIDA → Nordic Conference on Computational Linguistics
*SEM → Joint Conference on Lexical and Computational Semantics
SemEval → International Workshop on Semantic Evaluation
SP-Sem-MRL → Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages
TALN → French Conference on Natural Language Processing
TLT → International Workshop on Treebanks and Linguistic Theories
TSD → International Conference on Text, Speech and Dialogue
WDS → Week of Doctoral Students Conference
WMT → Workshop on Statistical Machine Translation

Bibliography

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Paşca, M., and Soroa, A. (2009). A study on similarity and relatedness using distributional and WordNet-based approaches. In *NAACL-HLT*.

Agirre, E., Baldwin, T., and Martínez, D. (2008). Improving Parsing and PP Attachment Performance with Sense Information. In *ACL*.

Agirre, E., Bengoetxea, K., Gojenola, K., and Nivre, J. (2011). Improving dependency parsing with semantic classes. In *HLT*.

Ammar, W., Dyer, C., and Smith, N. A. (2014). Conditional random field autoencoders for unsupervised structured prediction. In *NIPS*.

Atkins, S. B. T. and Rundell, M. (2008). *The Oxford guide to practical lexicography*. Oxford University Press.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet Project. In *COLING*.

Baker, S., Reichart, R., and Korhonen, A. (2014). An unsupervised model for instance level subcategorization acquisition. In *EMNLP*.

Bansal, M., Denero, J., and Lin, D. (2012). Unsupervised translation sense clustering. In *NAACL-HLT*.

Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. In *ACL*.

Barber, D. (2012). *Bayesian reasoning and machine learning*. Cambridge University Press.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*.

Baroni, M. and Lenci, A. (2010). Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. (2015). Breaking sticks and ambiguities with adaptive skip-gram. *arXiv preprint arXiv:1502.07257*.

Baum, L. E. (1972). An inequality and associated maximization technique in statistical estimation for probalistic functions of Markov processes. In *Inequalities*.

Belinkov, Y., Lei, T., Barzilay, R., and Globerson, A. (2014). Exploring compositional architectures and word vector representations for prepositional phrase attachment. *Transactions of the Association for Computational Linguistics*, 2:561–572.

Bender, E. (2013). *Linguistic Fundamentals for Natural Language Processing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Bender, E. M. (2011). On Achieving and Evaluating Language-Independence in NLP. *Linguistic Issues in Language Technology*, 6(3):1–26.

Bengio, Y. (1999). Markovian models for sequential data. *Neural computing surveys*, 2:129–162.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Bengio, Y. and Frasconi, P. (1996). Input-output HMMs for sequence processing. *IEEE Transactions on Neural Networks*, 7(5).

Bikel, D. M. (2000). A statistical model for parsing and word-sense disambiguation. In *EMNLP*.

Bikel, D. M. (2002). Design of a multi-lingual, parallel-processing statistical parsing engine. In *HLT*.

Bikel, D. M. (2004). Intricacies of collins' parsing model. *Computational Linguistics*, 30(4):479–511.

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition.

Blacoe, W. and Lapata, M. (2012). A comparison of vector-based representations for semantic composition. In *EMNLP-CONLL*.

Bohnet, B. (2010). Top accuracy and fast dependency parsing is not a contradiction. In *COLING*.

Bojar, O., Buck, C., Callison-Burch, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2013). Findings of the 2013 Workshop on Statistical Machine Translation. In *WMT*.

Bojar, O., Žabokrtský, Z., Dušek, O., Galuščáková, P., Majliš, M., Mareček, D., Maršík, J., Novák, M., Popel, M., and Tamchyna, A. (2012). The Joy of Parallelism with CzEng 1.0. In *LREC*.

Bouma, G., van Noord, G., and Malouf, R. (2001). Alpino: Wide Coverage Computational Analysis of Dutch. In *Computational Linguistics in the Netherlands 2000*.

Boyd-Graber, J. and Blei, D. M. (2008). Syntactic topic models. In *NIPS*.

Brants, T. (2000). TnT: a statistical part-of-speech tagger. In *ANLC*.

Brown, P. F., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. (1991). Word-sense disambiguation using statistical methods. In *ACL*.

Brown, P. F., Pietra, V. J. D., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Bruni, E., Boleda, G., Baroni, M., and Tran, N.-K. (2012). Distributional semantics in technicolor. In *ACL*.

Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *WMT*.

Candito, M. and Crabbé, B. (2009). Improving generative statistical parsing with semi-supervised word clustering. In *IWPT*.

Cappé, O. and Moulines, E. (2009). Online EM algorithm for latent data models. *Journal of the Royal Statistical Society*.

Carroll, J. and Briscoe, T. (1996). Apportioning development effort in a probabilistic lr parsing system through evaluation. In *EMNLP*.

Chandar A P, S., Lauly, S., Larochelle, H., Khapra, M. M., Ravindran, B., Raykar, V. C., and Saha, A. (2014). An autoencoder approach to learning bilingual word representations. In *NIPS*.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *NAACL*.

Charniak, E. (2001). Immediate-head parsing for language models. In *ACL*.

Charniak, E., Blaheta, D., Ge, N., Hall, K., Hale, J., and Johnson, M. (2000). *BLLIP 1987–1989 WSJ Corpus Release 1, LDC No. LDC2000T43*. Linguistic Data Consortium.

Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *EMNLP*.

Chen, W., Zhang, M., and Li, H. (2012). Utilizing dependency language models for graph-based dependency parsing models. In *ACL*.

Chen, X., Liu, Z., and Sun, M. (2014). A unified model for word sense representation and disambiguation. In *EMNLP*.

Christodouloupoulos, C. and Steedman, M. (2014). A massively parallel corpus: the bible in 100 languages. *Language Resources and Evaluation*, 49(2):375–395.

Chrupala, G. (2011). Efficient induction of probabilistic word classes with LDA. In *IJCNLP*.

Ciaramita, M. and Altun, Y. (2006). Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *EMNLP*.

Clark, S. (2001). *Class-Based Statistical Models for Lexical Knowledge Acquisition*. PhD thesis, University of Sussex.

Collins, M. (2002). Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *EMNLP*.

Collins, M. (2003). Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *ICML*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Coulmance, J., Marty, J.-M., Wenzek, G., and Benhalloum, A. (2015). Transgram, fast cross-lingual word-embeddings. In *EMNLP*.

Cruse, D. (1986). *Lexical Semantics*. Cambridge University Press, Cambridge, UK.

Daelemans, W. and van den Bosch, A. (2005). *Memory-based language processing*. Cambridge University Press.

Dagan, I. and Itai, A. (1994). Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics*, 20(4):563–596.

Dagan, I., Marcus, S., and Markovitch, S. (1993). Contextual word similarity and estimation from sparse data. In *ACL*.

Das, D., Chen, D., Martins, A. F. T., Schneider, N., and Smith, N. A. (2014). Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Denkowski, M., Hanneman, G., and Lavie, A. (2012). The CMU-Avenue French-English Translation System. In *WMT*.

Derczynski, L. and Chester, S. (2016). Generalised Brown Clustering and Roll-Up Feature Generation. In *AAAI*.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. (2014). Fast and robust neural network joint models for statistical machine translation. In *ACL*.

Diab, M. and Resnik, P. (2002). An unsupervised method for word sense tagging using parallel corpora. In *ACL*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Duda, R. O., Hart, P. E., et al. (1973). *Pattern classification and scene analysis*. Wiley New York.

Dyer, C., Lopez, A., Ganitkevitch, J., Weese, J., Ture, F., Blunsom, P., Setiawan, H., Eidelman, V., and Resnik, P. (2010). cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *ACL*.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., and Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In *NAACL-HLT*.

Faruqui, M. and Dyer, C. (2013). An information theoretic approach to bilingual word clustering. In *ACL*.

Faruqui, M. and Dyer, C. (2014a). Community evaluation and exchange of word vectors at wordvectors.org. In *ACL System Demonstrations*.

Faruqui, M. and Dyer, C. (2014b). Improving vector space word representations using multilingual correlation. In *EACL*.

Faruqui, M. and Dyer, C. (2015). Non-distributional word vector representations. In *ACL*.

Fellbaum, C. (1998). *WordNet*. MIT Press.

Fillmore, C. (1982). Frame semantics. *Linguistics in the morning calm*, pages 111–137.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2001). Placing search in context: The concept revisited. In *WWW*.

Firth, J. (1957). A synopsis of linguistic theory 1930-1955. *Studies in linguistic analysis*, pages 1–32.

Francis, W. N. and Kucera, H. (1979). Brown corpus manual. *Brown University*.

Frank, S., Goldwater, S., and Keller, F. (2013). Adding sentence types to a model of syntactic category acquisition. *Topics in Cognitive Science*, 5(3):495–521.

Fujita, S., Bond, F., Oepen, S., and Tanaka, T. (2007). Exploiting semantic information for HPSG parse selection. In *Proceedings of the Workshop on Deep Linguistic Processing*, DeepLP.

Ganchev, K., Graça, J., Gillenwater, J., and Taskar, B. (2010). Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049.

Ganchev, K., Graça, J. a. V., and Taskar, B. (2008). Better alignments = better translations? In *ACL-HLT*.

Garrette, D. and Baldridge, J. (2012). Type-Supervised Hidden Markov Models for Part-of-Speech Tagging with Incomplete Tag Dictionaries. In *EMNLP*.

Goldberg, Y. and Orwant, J. (2013). A dataset of syntactic-ngrams over time from a very large corpus of english books. In **SEM*.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep learning. Book in preparation for MIT Press.

Gouws, S., Bengio, Y., and Corrado, G. (2014). BilBOWA: Fast Bilingual Distributed Representations without Word Alignments. *arXiv preprint arXiv:1410.2455*.

Grave, E., Obozinski, G., and Bach, F. (2013). Hidden Markov tree models for semantic class induction. In *CoNLL*.

Grave, E., Obozinski, G., and Bach, F. (2014). A Markovian approach to distributional semantics with application to semantic compositionality. In *COLING*.

Graça, J., Ganchev, K., and Taskar, B. (2007). Expectation maximization and posterior constraints. In *NIPS*.

Greenberg, C., Sayeed, A., and Demberg, V. (2015). Improving unsupervised vector-space thematic fit evaluation via role-filler prototype clustering. In *NAACL*.

Grefenstette, G. (1994). *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers.

Gries, S. T. (2009). *Statistics for linguistics with R*. Mouton de Gruyter.

Guo, J., Che, W., Wang, H., and Liu, T. (2014). Learning sense-specific word embeddings by exploiting bilingual resources. In *COLING*.

Guo, J., Che, W., Yarowsky, D., Wang, H., and Liu, T. (2016). A representation learning framework for multi-source transfer parsing.

Haffari, G., Razavi, M., and Sarkar, A. (2011). An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *ACL*.

Halawi, G., Dror, G., Gabrilovich, E., and Koren, Y. (2012). Large-scale learning of word relatedness with constraints. In *KDD*.

Harabagiu, S., Miller, G., and Moldovan, D. (1999). Wordnet 2 - a morphologically and semantically enhanced resource. In *SIGLEX99: Standardising Lexical Resources*.

Harris, Z. (1954). Distributional structure. *Word*, 10(23):146–162.

Hearst, M. A. and Schütze, H. (1996). Customizing a lexicon to better suit a computational task. In *Workshop On The Acquisition Of Lexical Knowledge From Text*.

Henestroza Anguiano, E. and Candito, M. (2012). Probabilistic lexical generalization for French dependency parsing. In *SP-Sem-MRL*.

Hermann, K. M. and Blunsom, P. (2014). Multilingual models for compositional distributed semantics. In *ACL*.

Hermann, K. M., Das, D., Weston, J., and Ganchev, K. (2014). Semantic frame identification with distributed word representations. In *ACL*.

Hill, F., Cho, K., Jean, S., Devin, C., and Bengio, Y. (2014a). Embedding word similarity with neural machine translation. *arXiv preprint arXiv:1412.6448*.

Hill, F., Reichart, R., and Korhonen, A. (2014b). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *arXiv preprint arXiv:1408.3456*.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *ACL*.

Hirschberg, J. and Manning, C. D. (2015). Advances in natural language processing. *Science*, 349(6245):261–266.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *ACL*.

Huang, F., Ahuja, A., Downey, D., Yang, Y., Guo, Y., and Yates, A. (2014). Learning representations for weakly supervised natural language processing tasks. *Computational Linguistics*, 40(1):85–120.

Huang, F., Yates, A., Ahuja, A., and Downey, D. (2011). Language models as representations for weakly-supervised nlp tasks. In *CoNLL*.

Hürlimann, M., Weck, B., van den Berg, E., Šuster, S., and Nissim, M. (2015). GLAD: Groningen Lightweight Authorship Detection. In *Working Notes Papers of the CLEF 2015 Evaluation Labs*.

Ide, N. (2000). Cross-lingual sense determination: Can it work? *Computers and the Humanities*, 34(1-2):223–234.

Ide, N. and Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):2–40.

Johansson, R. and Nugues, P. (2007). Extended constituent-to-dependency conversion for English. In *NODALIDA*.

Jurafsky, D. and Martin, J. H. (2008). *Speech and Language Processing (2nd Edition)*. Prentice Hall, 2 edition.

Kaji, H. (2003). Word sense acquisition from bilingual comparable corpora. In *NAACL-HLT*.

Kawakami, K. and Dyer, C. (2016). Learning to represent words in context with multilingual supervision. In *ICLR Workshop Papers*.

Kiperwasser, E. and Goldberg, Y. (2015). Semi-supervised dependency parsing using bilexical contextual features from auto-parsed data. In *EMNLP*.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Skip-thought vectors. In *NIPS*.

Klein, D. (2005). *The unsupervised learning of natural language structure*. PhD thesis, Stanford University.

Klementiev, A., Titov, I., and Bhattarai, B. (2012). Inducing crosslingual distributed representations of words. In *COLING*.

Klusáček, D. (2006). Maximum mutual information and word classes. In *WDS*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *ACL-HLT*.

Krishnan, V. and Manning, C. D. (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *COLING-ACL*.

Kübler, S., McDonald, R. T., and Nivre, J. (2009). *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden Markov model. *Computer Speech & Language*, 6(3):225–242.

Landauer, T. K., McNamara, D. S., Dennis, S., and Kintsch, W. (2007). *Handbook of Latent Semantic Analysis*. Lawrence Erlbaum, Mahwah, New Jersey.

Larochelle, H., Erhan, D., Courville, A., Bergstra, J., and Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML*.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *ICML*.

Leech, G. (1992). 100 million words of english: the british national corpus (bnc). *Language Research*, 28(1):1–13.

Lember, J. and Koloydenko, A. A. (2014). Bridging Viterbi and posterior decoding: A generalized risk approach to hidden path inference based on Hidden Markov models. *Journal of Machine Learning Research*, 15(1):1–58.

Levin, B. (1993). *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press.

Levy, O. and Goldberg, Y. (2014a). Dependency-based word embeddings. In *ACL*.

Levy, O. and Goldberg, Y. (2014b). Linguistic regularities in sparse and explicit word representations. *CoNLL*.

Levy, O. and Goldberg, Y. (2014c). Neural word embedding as implicit matrix factorization. In *NIPS*.

Li, J. and Jurafsky, D. (2015). Do multi-sense embeddings improve natural language understanding? In *EMNLP*.

Liang, P. (2005). Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Liang, P. and Klein, D. (2009). Online EM for unsupervised models. In *HLT-NAACL*.

Lin, D. (1998a). Automatic retrieval and clustering of similar words. In *COLING*.

Lin, D. (1998b). An information-theoretic definition of similarity. In *ICML*.

Lin, D. (2003). Dependency-based evaluation of MINIPAR. In *Treebanks*, pages 317–329. Springer.

Lin, D. and Wu, X. (2009). Phrase clustering for discriminative learning. In *ACL-IJCNLP of AFNLP*.

Ling, W., Tsvetkov, Y., Amir, S., Fermandez, R., Dyer, C., Black, A. W., Trancoso, I., and Lin, C.-C. (2015). Not all contexts are created equal: Better word representations with variable attention. In *EMNLP*.

Lu, A., Wang, W., Bansal, M., Gimpel, K., and Livescu, K. (2015). Deep multilingual correlation for improved word embeddings. In *NAACL*.

Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers*, 28(2):203–208.

Luong, M.-T., Socher, R., and Manning, C. D. (2013). Better word representations with recursive neural networks for morphology. In *CoNLL*.

Luong, T., Pham, H., and Manning, C. D. (2015). Bilingual word representations with monolingual quality in mind. In *Workshop on Vector Space Modeling for NLP*.

MacKinlay, A., Dridan, R., McCarthy, D., and Baldwin, T. (2012). The effects of semantic annotations on precision parse ranking. In *SemEval*.

Manning, C. D. (2011). Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *Computational Linguistics and Intelligent Text Processing*, pages 171–189. Springer.

Manning, C. D. (2016). Computational linguistics and deep learning. *Computational Linguistics*.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA.

Marcheggiani, D. and Titov, I. (2016). Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–330.

Martin, W., Maks, I., Bopp, S., and Groot, M. (2005). Referentie bestand nederlands documentatie. Technical report, Free University Amsterdam.

McCarthy, D., Koeling, R., Weeds, J., and Carroll, J. A. (2004). Finding predominant word senses in untagged text. In *ACL*.

McDonald, R. and Pereira, F. (2006). Online learning of approximate dependency parsing algorithms. In *EACL*.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *HLT-EMNLP*.

Merialdo, B. (1994). Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–171.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In *ICLR Workshop Papers*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *NIPS*.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11).

Miller, G. A. and Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

Miller, S., Guinness, J., and Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *HLT-NAACL*.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc.

Mnih, A. and Hinton, G. (2007). Three new graphical models for statistical language modelling. In *ICML*.

Momtazi, S., Khudanpur, S., and Klakow, D. (2010). A comparative study of word co-occurrence for term clustering in language model-based sentence retrieval. In *ACL-HLT*.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Nalisnick, E. and Ravi, S. (2015). Infinite dimensional word embeddings. *arXiv preprint arXiv:1511.05392*.

Naseem, T., Snyder, B., Eisenstein, J., and Barzilay, R. (2009). Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 36:1–45.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *EMNLP*.

Nepal, A. and Yates, A. (2014). Factorial Hidden Markov models for learning representations of natural language. In *ICLR*.

Ng, H. T., Wang, B., and Chan, Y. S. (2003). Exploiting parallel texts for word sense disambiguation: An empirical study. In *ACL*.

Nivre, J. (2006). *Inductive Dependency Parsing (Text, Speech and Language Technology)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Oostdijk, N., Reynaert, M., Monachesi, P., van Noord, G., Ordelman, R., Schuurman, I., and Vandeghinste, V. (2008). From D-Coi to SoNaR: a reference corpus for Dutch. In *LREC*.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*.

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Computational Linguistics*, 33:161–199.

Pal, C., Sutton, C., and McCallum, A. (2006). Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *ICASSP*.

Passos, A., Kumar, V., and McCallum, A. (2014). Lexicon infused phrase embeddings for named entity resolution. In *CoNLL*.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Pedersen, T. (2010). Information content measures of semantic similarity perform better without sense-tagged text. In *NAACL-HLT*.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. *EMNLP*.

Pereira, F., Tishby, N., and Lee, L. (1993). Distributional clustering of english words. In *ACL*.

Petrov, S. (2009). *Coarse-to-Fine Natural Language Processing*. PhD thesis, University of California at Berkeley.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *ACL*.

Petrov, S., Das, D., and McDonald, R. (2012). A universal part-of-speech tagset. In *LREC*.

Pitler, E. (2012). Attacking parsing bottlenecks with unlabeled data and relevant factorizations. In *ACL*.

Plank, B. (2011). *Domain Adaptation for Parsing*. PhD thesis, University of Groningen.

Plank, B. and Moschitti, A. (2013). Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *ACL*.

Plank, B. and van Noord, G. (2010). Grammar-driven versus data-driven: Which parsing system is more affected by domain shifts? In *NLPLING Workshop*.

Popat, K., A.R, B., Bhattacharyya, P., and Haffari, G. (2013). The haves and the have-nots: Leveraging unlabelled corpora for sentiment analysis. In *ACL*.

Popel, M. and Mareček, D. (2010). Perplexity of n-gram and dependency language models. In *TSD*.

Postma, M., van Miltenburg, E., Segers, R., Schoen, A., and Vossen, P. (2016). Open Dutch WordNet. In *GWC*.

Qu, L., Ferraro, G., Zhou, L., Hou, W., Schneider, N., and Baldwin, T. (2015). Big Data Small Data, In Domain Out-of Domain, Known Word Unknown Word: The Impact of Word Representation on Sequence Labelling Tasks. *arXiv preprint arXiv:1504.05319*.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Radinsky, K., Agichtein, E., Gabrilovich, E., and Markovitch, S. (2011). A word at a time: computing word relatedness using temporal semantic analysis. In *WWW*.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *CoNLL*.

Reisinger, J. and Mooney, J. R. (2010). Multi-prototype vector-space models of word meaning. In *HLT-NAACL*.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*.

Resnik, P. and Yarowsky, D. (1999). Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. *Natural Language Engineering*, 5:113–133.

Roget, P. M. (1911). *Roget's Thesaurus of English Words and Phrases*. TY Crowell Company.

Rooth, M., Riezler, S., Prescher, D., Carroll, G., and Beil, F. (1999). Inducing a Semantically Annotated Lexicon via EM-based Clustering. In *ACL*.

Rubenstein, H. and Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Ruge, G. (1992). Experiments on linguistically-based term associations. *Information Processing & Management*, 28(3):317–332.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E., McClelland, J. L., and PDP Research Group, C., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, pages 318–362. MIT Press, Cambridge, MA, USA.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R., Johnson, C. R., and Scheffczyk, J. (2006). *FrameNet II: Extended Theory and Practice*. International Computer Science Institute.

Sagae, K. and Gordon, A. S. (2009). Clustering words by syntactic similarity improves dependency parsing of predicate-argument structures. In *IWPT*.

Sahlgren, M. (2006). *The Word-Space Model: Using Distributional Analysis to Represent Syntagmatic and Paradigmatic Relations between Words in High-Dimensional Vector Spaces*. PhD thesis, Stockholm University, Stockholm, Sweden.

Sánchez, D., Batet, M., and Isern, D. (2011). Ontology-based information content computation. *Knowledge-Based Systems*, 24(2):297–303.

Schnabel, T., Labutov, I., Mimno, D., and Joachims, T. (2015). Evaluation methods for unsupervised word embeddings. In *EMNLP*.

Séaghdha, D. and Korhonen, A. (2014). Probabilistic distributional semantics with latent variable models. *Computational Linguistics*, 40(3):587–631.

Shaoul, C. and Westbury, C. (2010). The Westbury Lab Wikipedia Corpus. `http://www.psych.ualberta.ca/~westburylab/downloads/westburylab.wikicorp.download.html`.

Shen, L., Xu, J., and Weischedel, R. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL*.

Shi, Y., Zhang, W.-Q., Liu, J., and Johnson, M. (2013). Rnn language model with word clustering and class-based output layer. *EURASIP Journal on Audio, Speech, and Music Processing*, (1).

Smith, N. A. (2011). Linguistic structure prediction. *Synthesis lectures on human langSuage technologies*, 4(2):1–274.

Smith, N. A. and Eisner, J. (2005). Contrastive estimation: Training log-linear models on unlabeled data. In *ACL*.

Smith, N. A. and Eisner, J. (2006). Annealing structural bias in multilingual weighted grammar induction. In *COLING-ACL*.

Snyder, B. and Barzilay, R. (2010). Climbing the Tower of Babel: Unsupervised Multilingual Learning. In *ICML*.

Søgaard, A., Johannsen, A., Plank, B., Hovy, D., and Martínez Alonso, H. (2014). What's in a p-value in NLP? In *CoNLL*.

Soyer, H., Stenetorp, P., and Aizawa, A. (2014). Leveraging monolingual data for crosslingual compositional word representations. *CoRR*, abs/1412.6334.

Stolcke, A. (2002). SRILM—an extensible language modeling toolkit. In *ICSLP*.

Stratonovich, R. L. (1960). Conditional markov processes. *Theory of Probability & Its Applications*, 5(2):156–178.

Sun, A., Grishman, R., and Sekine, S. (2011). Semi-supervised relation extraction with large-scale word clustering. In *HLT-ACL*.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In *HLT-NAACL*.

Tian, F., Dai, H., Bian, J., Gao, B., Zhang, R., Chen, E., and Liu, T.-Y. (2014). A probabilistic model for learning multi-prototype word embeddings. In *COLING*.

Titov, I. and Khoddam, E. (2015). Unsupervised induction of semantic roles within a reconstruction-error minimization framework. In *NAACL*.

Titov, I. and Klementiev, A. (2012a). A Bayesian approach to unsupervised semantic role induction. In *EACL*.

Titov, I. and Klementiev, A. (2012b). Crosslingual induction of semantic roles. In *ACL*.

Tjong Kim Sang, E. and Hofmann, K. (2009). Lexical patterns or dependency patterns: Which is better for hypernym extraction? In *CoNLL*.

Tsvetkov, Y., Faruqui, M., Ling, W., Lample, G., and Dyer, C. (2015). Evaluation of word vector representations by subspace alignment. In *EMNLP*.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *ACL*.

Turney, P. D. and Pantel, P. (2010). From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Vadas, D. and Curran, J. R. (2007). Adding Noun Phrase Structure to the Penn Treebank. In *ACL*.

van de Cruys, T. (2010). *Mining for Meaning: The Extraction of Lexico-semantic Knowledge from Text*. PhD thesis, University of Groningen.

van Noord, G. (2006). At Last Parsing Is Now Operational. In *TALN*.

van Noord, G. (2007). Using self-trained bilexical preferences to improve disambiguation accuracy. In *IWPT*.

van Noord, G. (2009a). Huge parsed corpora in LASSY. In *TLT*.

van Noord, G. (2009b). Learning efficient parsing. In *EACL*.

van Noord, G. (2010). Self-trained bilexical preferences to improve disambiguation accuracy. In *Trends in Parsing Technology: Dependency Parsing, Domain Adaptation, and Deep Parsing*, Text, Speech and Language Technology.

Vossen, P., Maks, I., Segers, R., van der Vliet, H., Moens, M.-F., Hofmann, K., Tjong Kim Sang, E., and de Rijke, M., editors (2013). *Cornetto: A Combinatorial Lexical Semantic Database for Dutch*. Springer.

Šuster, S. (2012). Resolving PP-attachment ambiguity in French with distributional methods. Master's thesis, University of Groningen and University of Lorraine.

Šuster, S. (2015). An investigation into language complexity of World-of-Warcraft game-external texts. *arXiv preprint arXiv:1502.02655*.

Šuster, S., Titov, I., and van Noord, G. (2016). Bilingual learning of multi-sense embeddings with discrete autoencoders. In *NAACL-HLT*.

Šuster, S. and van Noord, G. (2013). Semantic mapping for lexical sparseness reduction in parsing. In *ESSLLI'13 Workshop on Extrinsic Parse Improvement*.

Šuster, S. and van Noord, G. (2014). From neighborhood to parenthood: the advantages of dependency representation over bigrams in Brown clustering. In *COLING*.

Šuster, S., van Noord, G., and Titov, I. (2015). Word representations, tree models and syntactic functions. *arXiv preprint arXiv:1508.07709*.

Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305.

Wang, Y., Pakhomov, S., Ryan, J. O., and Melton, G. B. (2015). Domain adaption of parsing for operative notes. *Journal of biomedical informatics*, 54:1–9.

Widdows, D. (2003). Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In *NAACL-HLT*.

Wilks, Y. and Stevenson, M. (1998). Word sense disambiguation using optimised combinations of knowledge sources. In *COLING*.

Xiong, D., Li, S., Liu, Q., Lin, S., and Qian, Y. (2005). Parsing the Penn Chinese treebank with semantic knowledge. In *IJCNLP*.

Xu, C., Bai, Y., Bian, J., Gao, B., Wang, G., Liu, X., and Liu, T.-Y. (2014). Rc-net: A general framework for incorporating knowledge into word representations. In *CIKM*.

Yang, D. and Powers, D. M. W. (2006). Verb similarity on the taxonomy of wordnet. In *GWC*.

Yu, M. and Dredze, M. (2014). Improving lexical embeddings with semantic knowledge. In *ACL*.

Zhang, J., Liu, S., Li, M., Zhou, M., and Zong, C. (2014). Bilingually-constrained phrase embeddings for machine translation. In *ACL*.

Zipf, G. K. (1949). *Human behavior and the principle of least effort*. Addison-Wesley Press.

Zou, G. Y. (2007). Toward using confidence intervals to compare correlations. *Psychological methods*, 12(4).

Zou, W. Y., Socher, R., Cer, D., and Manning, C. D. (2013). Bilingual word embeddings for phrase-based machine translation. In *EMNLP*.

Nederlandse samenvatting

Natuurlijke-taalverwerking is de analyse van natuurlijke taal door middel van computationele methoden. Om een hoog niveau van het begrijpen van de natuurlijke taal in praktische toepassingen te bereiken is taalanalyse vereist op allerlei niveaus, waaronder de analyse van de betekenis van woorden: de lexicale semantiek. Centraal in dit proefschrift staan verschillende benaderingen voor het verkrijgen van representaties van de betekenis van woorden als wiskundige objecten. Deze benaderingen zorgen voor een efficiënte computationele behandeling en interpreteerbaarheid, en maken het mogelijk om complexere taalverwerkingstaken uit te voeren. Steeds terugkerende problemen bij de verwerking van natuurlijke taal zijn de nieuwe situaties en contexten die steeds weer voorkomen, ook wanneer computationele modellen worden gebouwd op basis van zeer grote hoeveelheden tekst. De rol van woordrepresentaties is om deze nieuwe situaties doeltreffend aan te pakken door overeenkomsten met eerder voorgevallen situaties te benutten.

In het proefschrift motiveren we het gebruik van woordrepresentaties in het eerste deel, en we geven een overzicht van de relevante theoretische kaders en bestaande evaluatiemiddelen om de effectiviteit van de resulterende modellen te kunnen toetsen. Daarna volgt de empirische studie. In hoofdstuk 5 onderzoeken we hoe lexicale informatie kan worden gebruikt door een zinsontleder voor het Nederlands. We onderzoeken in meer detail het gebruik van lexicale voorkeuren om betere syntactische analyses op te leveren. Omdat de lexicale voorkeuren worden bepaald voor woorden

in een syntactische relatie (zoals onderwerp of lijdend voorwerp), bepalen we door middel van statistische technieken voor welke syntactische relaties het incorporeren van lexicale voorkeuren het grootste effect heeft op de ontledingskwaliteit. Voor deze relaties vervangen we in een experiment de betreffende woorden door hun semantische klassen uit het Nederlandse Wordnet. Door zowel fijne als grove semantische klassen te gebruiken kunnen we aantonen dat onze aanpak betere ontledingen oplevert in een aantal gevallen, maar uiteindelijk verbetert de algehele nauwkeurigheid van de ontleder niet.

In de hoofdstukken 6 en 7 werken we met woordrepresentaties die niet uit een bestaande, door de mens ontworpen bron (zoals Wordnet) zijn verkregen maar geheel automatisch op basis van grote corpora worden bepaald. De belangrijkste bijdrage van hoofdstuk 6 en 7 is dat de syntactische relatie tussen woorden en de aard van de syntactische relatie tussen woorden nuttig kunnen zijn voor het verkrijgen van preciezere woordrepresentaties. Meer bepaald onderzoeken wij twee modellen. In het eerste model wordt de semantiek van het woord bepaald door zijn lidmaatschap van één van de clusters die op basis van de data automatisch zijn geleerd. Hierbij meten wij de kwaliteit van een woordcluster door de semantische overeenkomsten tussen woorden in hetzelfde cluster te vergelijken met de overeenkomsten tussen die woorden in Wordnet.

In het tweede model wordt de semantiek van het woord gezien als afhankelijk van de context waarbij dus verschillende contexten van hetzelfde woord tot verschillende semantische representaties kunnen leiden. Deze laatste eigenschap wordt bereikt door het gebruik van het framework van latente variabelen. Om tot een preciezer onderscheid tussen woorden te komen onderzoeken we in hoofdstuk 7 niet alleen of woorden in een syntactische relatie tot elkaar voorkomen, maar we kijken ook naar de aard van die syntactische relatie; we gebruiken dus gelabelde syntactische relaties. De resulterende woordrepresentaties worden getest in twee toepassingen en daarbij wordt het voordeel van het gebruik van gelabelde syntactische relaties bevestigd. In beide hoofdstukken stellen we vast dat het voordeel

van het gebruik van syntactische relaties afhankelijk is van de taal van het onderzoek. Uit onze resultaten blijkt dat het Nederlands meer gebaat is bij syntactische informatie dan het Engels.

In hoofdstuk 8 richten we ons op het leren van woordrepresentaties over de taalgrenzen heen. Om de informatie uit een andere taal mee te nemen stellen we een techniek voor die geïnspireerd is op neurale netwerken. In dit hoofdstuk nemen we ook de polysemie van woorden serieus door de introductie van verschillende representaties voor elk van de betekenissen van een woord. Dit bereiken we door te beginnen met willekeurige representaties voor elk van de betekenissen die we daarna op iteratieve wijze verfijnen op basis van corpora. De hoofdbijdrage hierbij is dat de informatie uit gealigneerde corpora (d.w.z. vertalingen) kan worden gebruikt om woordbetekenissen betrouwbaarder te selecteren, waardoor de kwaliteit van de resulterende representaties verbeterd wordt. We concluderen dat meertalige informatie bijzonder nuttig kan zijn voor het verkrijgen van goede woordrepresentaties. Al zijn de resultaten van het gebruik van deze woordrepresentaties voor het automatisch toekennen van woordsoorten niet eenduidig, uit de evaluatie van de woordrepresentaties op een aantal standaard datasets waarbij de modellen worden ingezet om woordparen te selecteren die semantische overeenkomst vertonen blijken de nieuwe representaties duidelijk beter te presteren.

Groningen dissertations in linguistics (GRODIL)

1. Henriëtte de Swart (1991). *Adverbs of Quantification: A Generalized Quantifier Approach.*
2. Eric Hoekstra (1991). *Licensing Conditions on Phrase Structure.*
3. Dicky Gilbers (1992). *Phonological Networks. A Theory of Segment Representation.*
4. Helen de Hoop (1992). *Case Configuration and Noun Phrase Interpretation.*
5. Gosse Bouma (1993). *Nonmonotonicity and Categorial Unification Grammar.*
6. Peter I. Blok (1993). *The Interpretation of Focus.*
7. Roelien Bastiaanse (1993). *Studies in Aphasia.*
8. Bert Bos (1993). *Rapid User Interface Development with the Script Language Gist.*
9. Wim Kosmeijer (1993). *Barriers and Licensing.*
10. Jan-Wouter Zwart (1993). *Dutch Syntax: A Minimalist Approach.*
11. Mark Kas (1993). *Essays on Boolean Functions and Negative Polarity.*
12. Ton van der Wouden (1994). *Negative Contexts.*
13. Joop Houtman (1994). *Coordination and Constituency: A Study in Categorial Grammar.*
14. Petra Hendriks (1995). *Comparatives and Categorial Grammar.*
15. Maarten de Wind (1995). *Inversion in French.*

16. Jelly Julia de Jong (1996). *The Case of Bound Pronouns in Peripheral Romance.*
17. Sjoukje van der Wal (1996). *Negative Polarity Items and Negation: Tandem Acquisition.*
18. Anastasia Giannakidou (1997). *The Landscape of Polarity Items.*
19. Karen Lattewitz (1997). *Adjacency in Dutch and German.*
20. Edith Kaan (1997). *Processing Subject-Object Ambiguities in Dutch.*
21. Henny Klein (1997). *Adverbs of Degree in Dutch.*
22. Leonie Bosveld-de Smet (1998). *On Mass and Plural Quantification: The case of French 'des'/'du'-NPs.*
23. Rita Landeweerd (1998). *Discourse semantics of perspective and temporal structure.*
24. Mettina Veenstra (1998). *Formalizing the Minimalist Program.*
25. Roel Jonkers (1998). *Comprehension and Production of Verbs in aphasic Speakers.*
26. Erik F. Tjong Kim Sang (1998). *Machine Learning of Phonotactics.*
27. Paulien Rijkhoek (1998). *On Degree Phrases and Result Clauses.*
28. Jan de Jong (1999). *Specific Language Impairment in Dutch: Inflectional Morphology and Argument Structure.*
29. H. Wee (1999). *Definite Focus.*
30. Eun-Hee Lee (2000). *Dynamic and Stative Information in Temporal Reasoning: Korean tense and aspect in discourse.*
31. Ivilin P. Stoianov (2001). *Connectionist Lexical Processing.*
32. Klarien van der Linde (2001). *Sonority substitutions.*
33. Monique Lamers (2001). *Sentence processing: using syntactic, semantic, and thematic information.*
34. Shalom Zuckerman (2001). *The Acquisition of "Optional" Movement.*
35. Rob Koeling (2001). *Dialogue-Based Disambiguation: Using Dialogue Status to Improve Speech Understanding.*
36. Esther Ruigendijk (2002). *Case assignment in Agrammatism: a cross-linguistic study.*
37. Tony Mullen (2002). *An Investigation into Compositional Features and*

Feature Merging for Maximum Entropy-Based Parse Selection.
38. Nanette Bienfait (2002). *Grammatica-onderwijs aan allochtone jongeren.*
39. Dirk-Bart den Ouden (2002). *Phonology in Aphasia: Syllables and segments in level-specific deficits.*
40. Rienk Withaar (2002). *The Role of the Phonological Loop in Sentence Comprehension.*
41. Kim Sauter (2002). *Transfer and Access to Universal Grammar in Adult Second Language Acquisition.*
42. Laura Sabourin (2003). *Grammatical Gender and Second Language Processing: An ERP Study.*
43. Hein van Schie (2003). *Visual Semantics.*
44. Lilia Schürcks-Grozeva (2003). *Binding and Bulgarian.*
45. Stasinos Konstantopoulos (2003). *Using ILP to Learn Local Linguistic Structures.*
46. Wilbert Heeringa (2004). *Measuring Dialect Pronunciation Differences using Levenshtein Distance.*
47. Wouter Jansen (2004). *Laryngeal Contrast and Phonetic Voicing: ALaboratory Phonology.*
48. Judith Rispens (2004). *Syntactic and phonological processing indevelopmentaldyslexia.*
49. Danielle Bougaïré (2004). *L'approche communicative des campagnes de sensibilisation en santé publique au Burkina Faso: Les cas de la planification familiale, du sida et de l'excision.*
50. Tanja Gaustad (2004). *Linguistic Knowledge and Word Sense Disambiguation.*
51. Susanne Schoof (2004). *An HPSG Account of Nonfinite Verbal Complements in Latin.*
52. M. Begoña Villada Moirón (2005). *Data-driven identification of fixed expressions and their modifiability.*
53. Robbert Prins (2005). *Finite-State Pre-Processing for Natural Language Analysis.*
54. Leonoor van der Beek (2005) Topics in Corpus-Based Dutch Syntax

55. Keiko Yoshioka (2005). *Linguistic and gestural introduction and tracking of referents in L1 and L2 discourse.*
56. Sible Andringa (2005). *Form-focused instruction and the development of second language proficiency.*
57. Joanneke Prenger (2005). *Taal telt! Een onderzoek naar de rol van taalvaardigheid en tekstbegrip in het realistisch wiskundeonderwijs.*
58. Neslihan Kansu-Yetkiner (2006). *Blood, Shame and Fear: Self-Presentation Strategies of Turkish Women's Talk about their Health and Sexuality.*
59. Mónika Z. Zempléni (2006). *Functional imaging of the hemispheric contribution to language processing.*
60. Maartje Schreuder (2006). *Prosodic Processes in Language and Music.*
61. Hidetoshi Shiraishi (2006). *Topics in Nivkh Phonology.*
62. Tamás Biró (2006). *Finding the Right Words: Implementing Optimality Theory with Simulated Annealing.*
63. Dieuwke de Goede (2006). *Verbs in Spoken Sentence Processing: Unraveling the Activation Pattern of the Matrix Verb.*
64. Eleonora Rossi (2007). *Clitic production in Italian agrammatism.*
65. Holger Hopp (2007). *Ultimate Attainment at the Interfaces in Second Language Acquisition: Grammar and Processing.*
66. Gerlof Bouma (2008). *Starting a Sentence in Dutch: A corpus study of subject- and object-fronting.*
67. Julia Klitsch (2008). *Open your eyes and listen carefully. Auditory and audiovisual speech perception and the McGurk effect in Dutch speakers with and without aphasia.*
68. Janneke ter Beek (2008). *Restructuring and Infinitival Complements in Dutch.*
69. Jori Mur (2008). *Off-line Answer Extraction for Question Answering.*
70. Lonneke van der Plas (2008). *Automatic Lexico-Semantic Acquisition for Question Answering.*
71. Arjen Versloot (2008). *Mechanisms of Language Change: Vowel reduction in 15th century West Frisian.*
72. Ismail Fahmi (2009). *Automatic term and Relation Extraction for Medi-*

cal Question Answering System.

73. Tuba Yarbay Duman (2009). *Turkish Agrammatic Aphasia: Word Order, Time Reference and Case.*
74. Maria Trofimova (2009). *Case Assignment by Prepositions in Russian Aphasia.*
75. Rasmus Steinkrauss (2009). *Frequency and Function in WH Question Acquisition. A Usage-Based Case Study of German L1 Acquisition.*
76. Marjolein Deunk (2009). *Discourse Practices in Preschool. Young Children's Participation in Everyday Classroom Activities.*
77. Sake Jager (2009). *Towards ICT-Integrated Language Learning: Developing an Implementation Framework in terms of Pedagogy, Technology and Environment.*
78. Francisco Dellatorre Borges (2010). *Parse Selection with Support Vector Machines.*
79. Geoffrey Andogah (2010). *Geographically Constrained Information Retrieval.*
80. Jacqueline van Kruiningen (2010). *Onderwijsontwerp als conversatie. Probleemoplossing in interprofessioneel overleg.*
81. Robert G. Shackleton (2010). *Quantitative Assessment of English-American Speech Relationships.*
82. Tim Van de Cruys (2010). *Mining for Meaning: The Extraction of Lexico-semantic Knowledge from Text.*
83. Therese Leinonen (2010). *An Acoustic Analysis of Vowel Pronunciation in Swedish Dialects.*
84. Erik-Jan Smits (2010). *Acquiring Quantification. How Children Use Semantics and Pragmatics to Constrain Meaning.*
85. Tal Caspi (2010). *A Dynamic Perspective on Second Language Development.*
86. Teodora Mehotcheva (2010). *After the fiesta is over. Foreign language attrition of Spanish in Dutch and German Erasmus Student.*
87. Xiaoyan Xu (2010). *English language attrition and retention in Chinese and Dutch university students.*

88. Jelena Prokić (2010). *Families and Resemblances.*
89. Radek Šimík (2011). *Modal existential wh-constructions.*
90. Katrien Colman (2011). *Behavioral and neuroimaging studies on language processing in Dutch speakers with Parkinson's disease.*
91. Siti Mina Tamah (2011). *A Study on Student Interaction in the Implementation of the Jigsaw Technique in Language Teaching.*
92. Aletta Kwant (2011). *Geraakt door prentenboeken. Effecten van het gebruik van prentenboeken op de sociaal-emotionele ontwikkeling van kleuters.*
93. Marlies Kluck (2011). *Sentence amalgamation.*
94. Anja Schüppert (2011). *Origin of asymmetry: Mutual intelligibility of spoken Danish and Swedish.*
95. Peter Nabende (2011). *Applying Dynamic Bayesian Networks in Transliteration Detection and Generation.*
96. Barbara Plank (2011). *Domain Adaptation for Parsing.*
97. Cagri Coltekin (2011). *Catching Words in a Stream of Speech: Computational simulations of segmenting transcribed child-directed speech.*
98. Dörte Hessler (2011). *Audiovisual Processing in Aphasic and Non-Brain-Damaged Listeners: The Whole is More than the Sum of its Parts.*
99. Herman Heringa (2012). *Appositional constructions.*
100. Diana Dimitrova (2012). *Neural Correlates of Prosody and Information Structure.*
101. Harwintha Anjarningsih (2012). *Time Reference in Standard Indonesian Agrammatic Aphasia.*
102. Myrte Gosen (2012). *Tracing learning in interaction. An analysis of shared reading of picture books at kindergarten.*
103. Martijn Wieling (2012). *A Quantitative Approach to Social and Geographical Dialect Variation.*
104. Gisi Cannizzaro (2012). *Early word order and animacy.*
105. Kostadin Cholakov (2012). *Lexical Acquisition for Computational Grammars.A Unified Model.*
106. Karin Beijering (2012). *Expressions of epistemic modality in Main-*

land Scandinavian. A study into the lexicalization-grammaticalization-pragmaticalization interface.

107. Veerle Baaijen (2012). *The development of understanding through writing.*
108. Jacolien van Rij (2012). *Pronoun processing: Computational, behavioral, and psychophysiological studies in children and adults.*
109. Ankelien Schippers (2012). *Variation and change in Germanic long-distance dependencies.*
110. Hanneke Loerts (2012). *Uncommon gender: Eyes and brains, native and second language learners, & grammatical gender.*
111. Marjoleine Sloos (2013). *Frequency and phonological grammar: An integrated approach. Evidence from German, Indonesian, and Japanese.*
112. Aysa Arylova (2013). *Possession in the Russian clause. Towards dynamicity in syntax.*
113. Daniël de Kok (2013). *Reversible Stochastic Attribute-Value Grammars.*
114. Gideon Kotzé (2013). *Complementary approaches to tree alignment: Combining statistical and rule-based methods.*
115. Fridah Katushemererwe (2013). *Computational Morphology and Bantu Language Learning: an Implementation for Runyakitara.*
116. Ryan C. Taylor (2013). *Tracking Referents: Markedness, World Knowledge and Pronoun Resolution.*
117. Hana Smiskova-Gustafsson (2013). *Chunks in L2 Development: A Usage-based Perspective.*
118. Milada Walková (2013). *The aspectual function of particles in phrasal verbs.*
119. Tom O. Abuom (2013). *Verb and Word Order Deficits in Swahili-English bilingual agrammatic speakers.*
120. Gülsen Yılmaz (2013). *Bilingual Language Development among the First Generation Turkish Immigrants in the Netherlands.*
121. Trevor Benjamin (2013). *Signaling Trouble: On the linguistic design of other-initiation of repair in English conversation.*
122. Nguyen Hong Thi Phuong (2013). *A Dynamic Usage-based Approach*

to Second Language Teaching.

123. Harm Brouwer (2014). *The Electrophysiology of Language Comprehension: A Neurocomputational Model.*
124. Kendall Decker (2014). *Orthography Development for Creole Languages.*
125. Laura S. Bos (2015). *The Brain, Verbs, and the Past: Neurolinguistic Studies on Time Reference.*
126. Rimke Groenewold (2015). *Direct and indirect speech in aphasia: Studies of spoken discourse production and comprehension.*
127. Huiping Chan (2015). *A Dynamic Approach to the Development of Lexicon and Syntax in a Second Language.*
128. James Griffiths (2015). *On appositives.*
129. Pavel Rudnev (2015). *Dependency and discourse-configurationality: A study of Avar.*
130. Kirsten Kolstrup (2015). *Opportunities to speak. A qualitative study of a second language in use.*
131. Güliz Güneş (2015). *Deriving Prosodic structures.*
132. Cornelia Lahmann (2015). *Beyond barriers. Complexity, accuracy, and fluency in long-term L2 speakers' speech.*
133. Sri Wachyunni (2015). *Scaffolding and Cooperative Learning: Effects on Reading Comprehension and Vocabulary Knowledge in English as a Foreign Language.*
134. Albert Walsweer (2015). *Ruimte voor leren. Een etnografisch onderzoek naar het verloop van een interventie gericht op versterking van het taalgebruik in een knowledge building environment op kleine Friese basisscholen.*
135. Aleyda Lizeth Linares Calix (2015). *Raising Metacognitive Genre Awareness in L2 Academic Readers and Writers.*
136. Fathima Mufeeda Irshad (2015). *Second Language Development through the Lens of a Dynamic Usage-Based Approach.*
137. Oscar Strik (2015). *Modelling analogical change. A history of Swedish and Frisian verb inflection.*
138. He Sun (2015). *Predictors and stages of very young child EFL learners' English development in China.*

139. Marieke Haan (2015). *Mode Matters. Effects of survey modes on participation and answering behavior.*
140. Nienke Houtzager (2015). *Bilingual advantages in middle-aged and elderly populations.*
141. Noortje Joost Venhuizen (2015). *Projection in Discourse: A data-driven formal semantic analysis.*
142. Valerio Basile (2015). *From Logic to Language: Natural Language Generation from Logical Forms.*
143. Jinxing Yue (2016). *Tone-word Recognition in Mandarin Chinese: Influences of lexical-level representations.*
144. Seçkin Arslan (2016). *Neurolinguistic and Psycholinguistic Investigations on Evidentiality in Turkish.*
145. Rui Qin (2016). *Neurophysiological Studies of Reading Fluency. Towards Visual and Auditory Markers of Developmental Dyslexia.*
146. Kashmiri Stec (2016). *Visible Quotation: The Multimodal Expression of Viewpoint.*
147. Yinxing Jin (2016). *Foreign language classroom anxiety: A study of Chinese university students of Japanese and English over time.*
148. Joost Hurkmans (2016). *The Treatment of Apraxia of Speech. Speech and Music Therapy, an Innovative Joint Effort*
149. Franziska Köder (2016). *Between direct and indirect speech: The acquisition of pronouns in reported speech.*
150. Femke Swarte (2016). *Predicting the mutual intelligibility of Germanic languages from linguistic and extra-linguistic factors.*
151. Sanne Kuijper (2016). *Communication abilities of children with ASD and ADHD. Production, comprehension, and cognitive mechanisms.*
152. Jelena Golubović (2016). *Mutual intelligibility in the Slavic language area.*
153. Nynke van der Schaaf (2016). *Kijk eens wat ik kan! Sociale praktijken in interactie tussen kinderen van 4-8 jaar in de buitenschoolse opvang.*
154. Simon Šuster (2016). *Empirical studies on word representations.*

Grodil
Center for Language and Cognition Groningen (CLCG)
P.O. Box 716
9700 AS Groningen
The Netherlands